Estimating expected exposure profiles using biplot interpolation

Francesca van Niekerk

Assignment presented in the partial fulfilment
of the requirement for the degree of
MCom (Financial Risk Management)
at the University of Stellenbosch

Supervisor: Dr. C.J. van der Merwe

March 2020

PLAGIARISM DECLARATION

1. Plagiarism is the use of ideas, material and other intellectual property of another's work and to present it as my own.
2. I agree that plagiarism is a punishable offence because it constitutes theft.
3. Accordingly, all quotations and contributions from any source whatsoever (including the internet) have been cited fully. I understand that the reproduction of text without quotation marks (even when the source is cited) is plagiarism.
4. I also understand that direct translations are plagiarism.
5. I declare that the work contained in this assignment, except otherwise stated, is my original work and that I have not previously (in its entirety or in part) submitted it for grading in this assignment or another assignment.

ACKNOWLEDGEMENTS

Firstly, I would like to express my gratitude to my supervisor, Dr. C.J. van der Merwe, for his guidance and comments throughout the course of this assignment.

Furthermore, I am thankful to my parents for their endless patience and support. Lastly, I am also thankful to my siblings and friends for their words of encouragement.

ABSTRACT

The accounting standards provides guidelines on how to determine the fair value for a financial asset or liability held at fair value. When considering the fair value of derivative instruments, some additional adjustments need to be made for counterparty credit risk. For interest rate swaps, in particular, one needs to calculate the effective exposure of the swap in order to make these adjustments. One of the most popular methods, albeit computationally intensive, is to calculate these exposures through Monte Carlo simulation. In this study an alternative method of calculating the effective exposure using biplot interpolation is proposed. In this proposed method, an analytical approach in approximating the effective exposure profile is implemented through fitting a beta function. The parameters for this beta function are then estimated through biplot interpolation, which in turn approximates the exposure profile. When the performance of the biplot interpolation approach was tested using a standard interval testing approach, the approximated biplot interpolated profile provided a reasonable approximation of the true profile.

OPSOMMING

Die rekeningkundige standaarde bevat riglyne vir die bepaling van die billike waarde van 'n finansiële bate of las wat teen billike waarde gehou word. Wanneer die billike waarde vir afgeleide instrumente beskoud word, moet verdere aanpassings aangebring word vir die teenparty-kredietrisiko. Vir rentekoersruilkontrakte moet die effektiewe blootstelling van die ruilkontrak bereken word om hierdie aanpassings moontlik te kan aan bring. Een van die gewildste metodes, hoewel berekeningsintensief, is om hierdie blootstellings met behulp van Monte Carlo-simulasie te bereken. In hierdie studie word 'n alternatiewe metode vir die berekening van die effektiewe blootstelling met behulp van bi-stipping-interpolasie voorgestel. In hierdie voorgestelde metode word 'n analitiese benadering gebruik om die effektiewe blootstellingsprofiel te benader deur 'n beta-funksie toe te pas. Die parameters vir hierdie beta-funksie word dan bepaal deur bi-stipping-interpolasie, wat op sy beurt die blootstellingsprofiel benader. Toe die uitvoering van die bi-stipping-interpolasiebenadering met behulp van 'n standaard-interval-toetsbenadering getoets is, het die benaderde bi-stipping-geïnterpoleerde profiel 'n redelike benadering van die ware profiel gegee.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND/OR ACRONYMS

AUC	Area Under Curve
cdf	Cumulative Distribution Function
CVA	Credit Value Adjustment
EE	Expected Exposure
IFRS	International Financial Reporting Standards
IRS	Interest Rate Swap
JIBAR	Johannesburg Interbank Average Rate
LGD	Loss-Given Default
MC	Monte Carlo
NS	Nelson-Siegel
OTC	Over-The-Counter
PCA	Principal Component Analysis
PD	Probability of Default
pdf	probability distribution function

LIST OF NOTATION

μ_V	mean of the future portfolio values
$\Phi(.)$	normal cumulative distribution function (cdf)
σ_V	annual standard deviation of the future portfolio values
$\phi(.)$	normal probability density function (pdf)
Not	notional value
r_{fix}	fixed swap rate
s	the spread above the reference interest rate
pmt_{freq}	payment frequency
r_{reset}	previous rest reference rate
T	maturity
L	level parameter in Nelson-Siegel model
S	slope parameter in Nelson-Siegel model
C	curve parameter in Nelson-Siegel model
λ	scale parameter in Nelson-Siegel model
δ	length in time
$r_{forward}$	forward rate
DF	discount factor
r	instantaneous short rate
$\theta(t)$	function of time used in the Hull-White model
a	mean reversion rate parameter in the Hull-White model
σ	volatility parameter in the Hull-White model
dz	Wiener process
N_{sims}	number of simulations
α	alpha parameter in the beta function
β	beta parameter in the beta function
r_{dim}	number of dimensions
n	number of observations
p	number of variables
X: $n \times p$	an input matrix containing n samples of p variables
Y: $n \times p$	an output matrix containing n samples of p variables
Z	scaffolding matrix
V	eigen matrix
P	percentile

CHAPTER 1

INTRODUCTION

The ability to understand and identify patterns within large data sets is of ever-growing importance as the amount of data accumulated in modern systems compounds. To make decisions based on collected data, one needs to be able to interpret the data correctly. Graphical tools play an important role in interpreting data, as trends in the data can be recognised easily. When considering a multivariate data set, a biplot representation of the data is an ideal tool for visualisation of the data, where both the observations and the variables of the multivariate data set are displayed in a single plot.

The valuation of financial instruments is often complex and dependent on a set of input variables. Various risk management measures are also calculated along with the valuation of these financial instruments. In this study, interest rate swaps (IRSs) and the subsequent estimation of expected exposure (EE) are studied. The EE of an IRS is calculated as the average positive exposure of the IRS over the maturity of the contract. The EE is calculated in order to determine the level of credit value adjustments (CVA) which need to be made to measure the fair value of an asset. The measurement of the fair value of assets is defined by the guidelines as set out by the International Financial Reporting Standards (IFRS) 13, and it is required to be calculated for accounting reporting purposes. The estimation of EE values therefore play an important role in the risk management of IRSs.

A Monte Carlo (MC) simulation approach, which is commonly used to estimate EE, is considered. The MC estimate of the EE profile is calculated as the average of the future exposure values of the IRS calculated over a large number of simulations. The underlying reference interest rate curve is used to determine the floating rate payments in an IRS. These usually have fixed payments that are netted against floating payments, and are discounted at the risk-free rate. In the MC simulation approach the underlying reference interest rate curve is simulated to estimate the exposure at future points in time. Next, a parameterisation of the resulting estimated EE profile calculated using the MC simulation approach, is proposed. In order to simplify the resource intensive MC simulation approach, the EE profile will be parameterised using some function. The estimated function parameters will then in turn be used to estimate the EE profile. A beta function, which is characterised by three parameters, is proposed as a suitable fit for the EE profile. The aim of the parameterisation of the EE profile is to estimate the EE profile for a given IRS using beta function parameters.

In this study, a grid of input values for IRSs is created (hereafter, referred to as the input grid) and the corresponding beta function parameters are estimated for each of these swaps in the grid. This is done by firstly estimating the EE profile for each swap, and thereafter fitting the beta function to the simulated profile, in order to obtain the relevant parameters of the beta function. These estimated beta function parameters for each IRS, are then recorded and stored in an output grid. These parameters, therefore, correspond to each of the various IRSs contained in the input grid. A biplot interpolation method is then proposed to link the input and output grids (training grids), in order to estimate the beta function parameters which will be used to estimate the EE profiles for IRSs, given a set of new input parameters.

The above biplot interpolation method aims to provide a simplified approach to EE estimation compared to the commonly used and computationally expensive MC simulation approach. While the MC simulation approach increases accuracy due to its flexibility, it comes at a price, as more complex computations will be required to run the simulations, which would in turn be more resource intensive. As an alternative, the biplot interpolation approach provides the end-user with a method which is easy and quick to implement, whilst still managing to provide a reasonable approximation of the EE profile. In this study, only non-amortising IRSs in which fixed-for-floating payments are exchanged, where payments are netted between parties. Further, the EE profiles will always be calculated from the point of the fixed rate payer. These calculations can, however, easily be changed to be from the point of the floating rate payer.

The outline of this study follows as: an overview on EE calculations, and the importance of EE calculations for risk management purposes, is covered in chapter 2. In chapter 3 principal compenents analysis (PCA) biplots are discussed, including the construction and interpretation thereof. The new biplot interpolation methodology is discussed in detail in chapter 4. Here an application of the biplot interpolation method for EE data is provided, and the methodology of the generation of the EE data used to apply the biplot interpolation method is discussed. The data generated in this chapter are the input and the output grids which will be used as training data in the biplot interpolation method.

Next, in chapter 5, new test input data are chosen at random and used to test the biplot interpolation method. The biplot interpolation method uses the training data generated in chapter 4 in calculations to estimate parameter values for the new test input data. The resulting EE values, calculated using the estimated parameters, are compared to the true EE values to test whether the biplot interpolated values provide reasonable estimates for the true EE. The results from this study are then reported, discussed, and concluded in chapter 6.

CHAPTER 2

EXPECTED EXPOSURE

In this chapter, the theory required to estimate the EE of IRSs is discussed. It is assumed that the reader has at least some primary understanding of of financial instrument valuation. The chapter begins with the theory of IRSs and the valuation thereof. Thereafter, EE is defined and the methodology used to calculate EE using MC simulations is discussed. A beta function is proposed as a fit for the EE profile, and the steps in estimating the beta function parameters to fit the EE profile will be described in the final section.

2.1 EXPECTED EXPOSURE METHODS

There are various methods which can be used to estimate the EE of an IRS. MC simulation is a popular method in which the positive exposure at future points in time is calculated over a large number of simulations and the values are aggregated to determine an estimate for the EE. In this study, an analytical approach is used, and below follows some examples of alternative analytical approaches discussed in the literature.

Gregory (2015) proposes an analytical approach, where the EE profile for an IRS is approximated using the assumption of normally distributed future portfolio values, i.e. a portfolio with future value, V, which is assumed to be normally distributed with mean, μ_V, and standard deviation, σ_V. Then, the EE values for an IRS can be derived at time t as: $EE(t) = E\big[\max(V, 0)\big] = \mu_V\, \Phi(\frac{\mu_V}{\sigma_V}) + \sigma_V\, \phi(\frac{\mu_V}{\sigma_V})$, where $\Phi(.)$ is the cumulative normal distribution function (cdf), and $\phi(.)$ is the normal probability density function (pdf).

Antwi (2014) studied the distribution of a loan portfolio, which is similar to the EE profile in that positions in a loan portfolio are also aggregated over a large number of simulations of positive and negative exposures. In the study, a MC simulation approach is used to determine the distribution of the loss distribution. Thereafter, an analytical approximation for the resulting distribution is derived using a known probability distribution. The beta distribution is proposed in their study and the fit is tested using goodness-of-fit tests.

An approach similar to Antwi (2014) will be followed in this research, where the methodology of fitting a function to a simulated distribution will be applied to IRS data. In particular, a beta function will be fitted to the simulated EE profile of the IRS data considered. In the following sections the theory used to simulate the EE values using MC simulation, and the fit of a beta function to the EE profile, will be discussed.

2.2 INTEREST RATE SWAPS

In this study, EE profiles are estimated for a predetermined input grid of IRSs. In order to estimate EE values, it is necessary to define the underlying IRS together with the parameters and rates used in valuing the swap. This is detailed in the following subsection.

2.2.1 Valuation

As already noted in the introductory chapter, the IRSs considered are fixed-for-floating IRSs. In these swaps, fixed interest rate payments are exchanged for floating rate payments plus a spread at periodic intervals on a notional amount. In such a swap, the reference floating rate is usually linked to a quoted interest rate in the market. In South Africa, most swaps use the three month Johannesburg Interbank Agreed Rate (JIBAR) as its reference.

Before discussing the valuation of the IRS, however, the following inputs used in the IRS are defined, namely the terms of the contract, and the current zero curve. Firstly, let $T_0 \leq T_1 \leq ... \leq T_n$ denote the payment times, with the first exchange of cash flows is at time T_1. The terms of the contract include the notional value of the contract, Not, the maturity of the contract, T_n, the fixed rate of the contract, r_{fix}, the spread above the reference interest rate, s, the payment frequency of the swap payments, pmt_{freq}, and the previous reset reference interest rate, r_{reset}. The current zero coupon risk-free rate is used for both the discounting and estimation of forward rates for the reference rate. For the exact valuation of an IRS, the true zero curve is used. The zero curve is often parameterised using various methods, with the most popular method being the Nelson-Siegel model. In this study, the Nelson-Siegel approximation is implemented, and the model is defined as (Nelson and Siegel, 1987):

$$y_t(\tau) = L + S\left(\frac{1-\exp(-\lambda\tau)}{\lambda\tau}\right) + C\left(\frac{1-\exp(-\lambda\tau)}{\lambda\tau} - \exp(-\lambda\tau)\right), \tag{2.1}$$

where L is defined as the level parameter, S is defined as the slope parameter, C is defined as the curvature parameter, and λ is the scale parameter which is usually chosen as a fixed value.

The value of the swap, $V(t)$, at time t , is now defined from the view point of a fixed rate payer, where the value of the swap is calculated as the difference between the value of the fixed leg payments and the value of the floating leg payments:

$$V(t) = V_{fixed}(t) - V_{float}(t). \tag{2.2}$$

It can be seen, that the above valuation can easily be changed to be from the view point of a floating rate payer, where the value is then simply calculated as $-V(t)$. The present value of the fixed leg in the swap is calculated as the sum of the discounted cash flows:

$$V_{fixed}(t) = \sum_{i=1}^{n} r_{fix} \cdot Not \cdot \delta_i \cdot DF(t, T_i), \tag{2.3}$$

where r_{fix} is the fixed rate, Not is the notional amount, $\delta_i = T_i - T_{i-1}$ is the length in years between consecutive payment dates, and $DF(t, T_i)$ is the discount factor, which is the factor used to discount a unit value from time T_i to time t, measured at time t.

Similarly, the present value of the floating leg in the swap contract, can be calculated as:

$$V_{float}(t) = \sum_{i=1}^{n} r_{forward}(t; T_{i-1}, T_i) \cdot Not \cdot \delta_i \cdot DF(t, T_i), \tag{2.4}$$

where $r_{forward}(t; T_{i-1}, T_i)$ is the forward rate, which applies to the period from time T_{i-1} to time T_i, measured at time t, and all the other parameters are the same as previously defined.

Now, the valuation of IRSs discussed in this subsection will now be used in the following subsection with the calculation of EE values.

2.2.2 Expected exposure

The exposure, that one of the counterparties have with regards to an IRS over the lifetime of the contract, is calculated by considering the future IRS values over the contract. The future IRS values are dependent on the underlying term structure of interest rates which is used to value the contract. The theory used to model the term structure of interest rates will now be discussed.

Under risk neutral pricing, the process of the short rate, r, is used to calculate the discount factors used to discount the cash flows in the IRS. These discount factors are in turn used to also calculate the forward rates used in estimating the floating rate payments. There are various models which are allow for the simulation of the short rate. Within the EE function (*CalcSimulatedExposure*) in the *xVA* package (Grivas, 2016), the Hull-White model (Hull and White, 1994) is used. The no-arbitrage Hull-White model uses the current zero curve as input. Because these models are calibrated to current market conditions it results in a no-arbitrage price.

The Hull-White model uses the following three inputs, namely, the initial term structure of interest rates (which will subsequently be parameterised by the Nelson-Siegel (NS) model), and the constant parameters: the mean reversion rate, a, and the volatility, σ. The Hull-White model for the term structure of interest rates is defined as (Hull and White, 1994):

$$dr = [\theta(t) - ar]\, dt + \sigma dz, \tag{2.5}$$

or

$$dr = a\left[\frac{\theta(t)}{a} - r\right] dt + \sigma dz, \tag{2.6}$$

where $\theta(t)$ is a function of time, a is the mean reversion rate, σ is the volatility, and dz is the Wiener process.

The function of time, $\theta(t)$, is derived, using the initial term structure as an input in the model, as:

$$\theta(t) = F_t(0,t) + aF(0,t) + \frac{\sigma^2}{2a}(1 - e^{2at}). \tag{2.7}$$

Now, the zero-coupon bond prices can be written as a function of the short rate at time t, and the prices at time t as follows:

$$P(t,T) = A(t,T)e^{-B(t,T)r(t)}, \tag{2.8}$$

where

$$B(t,T) = \frac{1 - e^{-a(T-t)}}{a}, \tag{2.9}$$

and

$$lnA(t,T) = ln\frac{P(0,T)}{P(0,t)} + B(t,T)F(0,t) - \frac{1}{4a^3}\sigma^2(e^{-aT} - e^{-at})^2(e^{2at} - 1). \tag{2.10}$$

The values $P(t,T)$ derived from the model can be used as the discount factor curve. This in turn is used to calculate the zero and forward rates needed in discounting the cash flows in the IRS at future points in time.

The MC simulation approach uses the above process by simulating different curves that are used in order to calculate the future exposure values of the IRS. The EE for the IRS is then estimated as follows:

$$\hat{EE}_{MC}(t) = \frac{1}{N_{sims}} \sum_{i=1}^{N_{sims}} E_i(t) = \frac{1}{N_{sims}} \sum_{i=1}^{N_{sims}} \max\left(0, V_i(t)\right), \tag{2.11}$$

where N_{sims} is the number of simulations, $E_i(t)$ is the exposure at time t, and $V_i(t)$ is the value of the IRS at time t as defined in the previous section.

2.2.3 CVA calculations

Trading derivatives in the over-the-counter (OTC) market, where contracts are not standardised as those traded on an exchange, the counterparties involved are consequently exposed to counterparty credit risk (CCR). The importance of managing counterparty credit risk was highlighted in the financial credit crisis of 2007, where mark-to-market losses on contracts were significantly high. This led to the inclusion of accounting for these losses through adjustments for credit risk.

In order to calculate the CVA for an IRS contract, the loss-given default (LGD), probability of default (PD), and EE need to be estimated. The CVA can be approximated at time t_i, with $i = 1, 2, ..., T$, for a contract which matures at time T as:

$$CVA(t_i) = LGD(t_i) \times Exposure(t_i) \times PD(t_i) \times DF(t_i, T) \tag{2.12}$$

There are two ways in which the CVA adjustment could be calculated. In the first approach, the CVA adjustment is calculated at each future simulation date at which the exposure is estimated, and thereafter aggregated. Alternatively, the EE profile can be simulated, and the CVA adjustment can be calculated on the EE profile, i.e. the CVA adjustment is only calculated once, after all the future exposure simulations have been aggregated.

As seen in the approximation of CVA, the estimation of EE is one of the main components. Therefore, in managing and quantifying counterparty credit risk, the estimation of exposure profiles for derivative contracts is a fundamental step.

In the following section, a beta function is proposed as a suitable fit to the simulated EE profile discussed in this section. The estimation of the beta function parameters are also discussed.

2.3 CURVE FITTING USING BETA FUNCTIONS

In the previous section, the MC simulation approach used to estimate the EE profile was discussed. A function will now be fitted to this EE profile. The chosen function is one which has flexibility in shape, in order to capture the various EE profiles which could be realised, given various IRS input parameters. The function which best fits the EE profile will be taken as a proxy for the unknown EE profile. This approach allows for new EE profiles to be determined given only the parameters of the beta function.

In the following subsection, the shape of the EE profile and the factors which affect the shape of the EE profile, are discussed. Thereafter, the parameters which determine the shape of the beta function and the fit of the beta function to EE profiles, are discussed.

2.3.1 The shape of the EE profile

The shape of the EE distribution provides an indication of the risk profile of the IRS contract. Gregory (2015) states that the maturity of the contract, the frequency and timing of the payments, optionality and default, which affect the shape of the EE profile. The optionality and default factors are beyond the scope of this study, and only symmetrical netted payments are considered. The two main factors which affect the shape of symmetrically netted contracts are the diffusion and amortisation effects (Zhu and Pykhtin, 2007).

The amortisation effect is the effect that, as the time to maturity of the contract decreases, the number of remaining payments to be exchanged decreases, which subsequently results in a decrease in the exposure. The diffusion effect is a result of the future uncertainty of the next payment due. As the time to the next payment date increases, the variability of the value of the payment due increases, this is due to the uncertainty of the future market realised interest rates. Therefore, the increased variability as the time to the next payment increases results in an increase in the exposure. The net effect of the opposing diffusion and amortisation effects over time, determines the shape of the EE profile.

The market conditions also affect the shape of the EE profile (Zhu and Pykhtin, 2007). The various yield curve scenarios: upward-sloping, inverted and hump shaped yield curves should be considered when estimating the parameters of the EE profiles.

2.3.2 The beta function

The beta function is proposed as a fit for the EE profile due to its flexibility in shape and because it is a bounded function. The beta function is characterised by the following parameters, namely a constant parameter, $Const$, a shape parameter, α, a curve parameter, β, and a boundary parameter, T. The function, $f_{AUC,\alpha,\beta,T}(x)$, is used to denote the beta function which will be used to fit the EE profile. The function, $f_{AUC,\alpha,\beta,T}(x)$, is defined on the interval $(0, T)$, where T is the maturity of the IRS, as:

$$f_{AUC,\alpha,\beta,T}(x) = Const \times x^{(\alpha-1)}(T - x)^{(\beta-1)}. \tag{2.13}$$

In the above function, the $Const$ parameter can be rewritten as $Const = AUC \times b(\alpha, \beta)$, where (AUC) is the area under curve, and $b(\alpha, \beta)$ is the Euler integral. The Euler integral, $b(\alpha, \beta)$, is defined as (Abramowitz and Stegun, 1948):

$$b(\alpha, \beta) = \int_0^1 t^{\alpha-1}(1 - t)^{\beta-1} dt. \tag{2.14}$$

Therefore, the function in equation 2.13, can be rewritten as:

$$f_{AUC,\alpha,\beta,T}(x) = AUC \times b(\alpha, \beta) \times x^{(\alpha-1)}(T - x)^{(\beta-1)}, \tag{2.15}$$

where the parameter estimates AUC, $\hat{\alpha}$, and $\hat{\beta}$ are solved using the nonlinear weighted least-squares estimation in R. The function, $f_{(AUC,\alpha,\beta,T)}(x)$, is parameterised in the equation above to capture the shape as well as the units of the EE profile.

The interval for the fitted beta function, is chosen for the interval $(0, T)$, which corresponds to the interval over which the EE of the swap contract is evaluated. The beta function is chosen as it is flexible, i.e. the α and β parameters, which determine the shape and curvature of the function, can be estimated to capture the majority of the shapes of the EE curve.

In figure 2.1 an example of a fitted beta function is plotted. The EE profile for a 10Y IRS is calculated and a beta function is fit to the profile. The estimated beta function parameters were calculated as: $AUC = 2.6198,\ ,\alpha = 1.2582,$ and $\beta = 1.6105$. The EE profile estimated, using the fitted beta function parameters, is plotted alongside the true EE profile, as seen in figure 2.1.

2.4 SUMMARY

Approximating EE values is one of the main aims of this study. This chapter provided the background theory used to estimate EE values. The chapter concluded with a section on curve fitting where a beta function was proposed as a fit for the EE profile. In the next chapter a brief overview of biplots will be given. The fitted EE profile will be used together with the biplots to derive the biplot interpolation method used to approximate EE profiles.

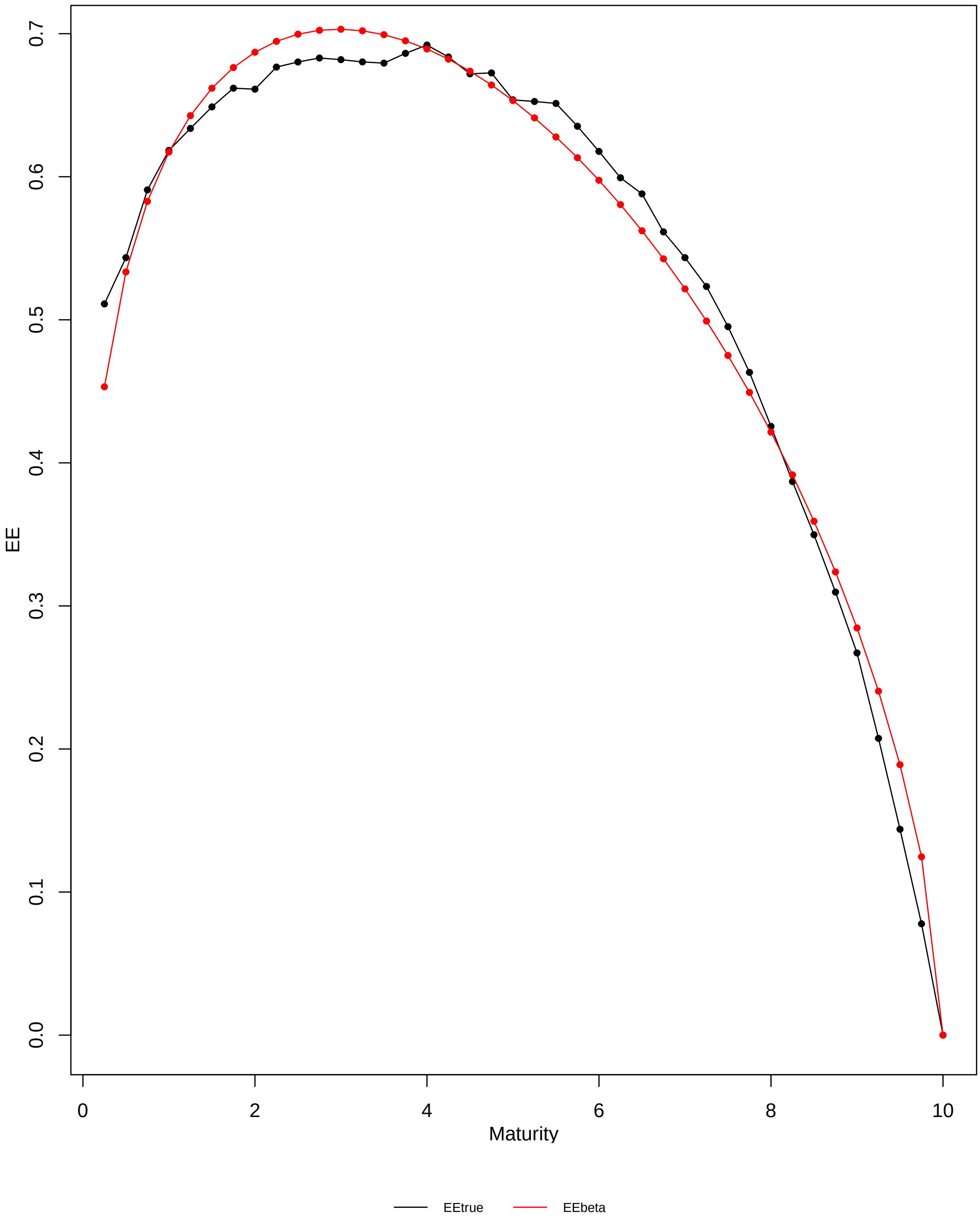


Figure 2.1: EE with beta fit

The *EEtrue* profile is the EE profile plotted for a 10Y IRS. A beta function is fitted to the EE profile, the estimated beta function parameters are: $AUC = 2.6198,\ ,\alpha = 1.2582,$ and $\beta = 1.6105.$ The *EEbeta* profile is the estimated EE profile using the fitted beta function parameter estimates.

CHAPTER 3

PRINCIPAL COMPONENTS ANALYSIS BIPLOTS

In the previous chapter, methods used to estimate EE were discussed. In the biplot interpolation approach, to be introduced in chapter 4, PCA biplots are used in the intermediary steps in estimating EE. This chapter provides the necessary theory behind the construction and interpretation of PCA biplots. In section 3.1, the construction and interpretation of biplots are discussed through a scatter plot analogy. The section starts with a discussion of the traditional two-dimensional scatter plot with two variables, after which the plot is extended to include observations with more than two variables.

In Le Roux *et al.* (2003), an argument for using biplots as a tool to describe multivariate variation and patterns in data is proposed. Specifically, in their paper they present examples related to financial data. A financial risk management example originally presented by Van Blerk, Gardner and le Roux (1999), is discussed. In this example, value-at-risk (VaR) calculations are graphically studied using PCA biplots. The other example covers an application of solvency data using canonical variate analysis biplots, which was originally studied by Oliver (1990). Canonical variate analysis biplots are used to describe data that can be divided into different classes. The study of canonical variate analysis biplots are beyond the scope of this study, as the data that will be used do not require classification.

3.1 PCA BIPLOTS

In many financial risk management calculations the data used and generated is both multivariate and quantitative. In this study, a multivariate input data set of expected exposure input parameters will be used. Biplots are, therefore, an ideal technique to display this data. The traditional biplot (Gabriel, 1971) displays the observations and variables in a two-dimensional display using points and arrows, where the points represent the observations and the arrows represent the variables. This biplot representation is not easy to interpret as the distances in the biplot are measured in terms of inner products (Le Roux *et al.*, 2003).

Gower and Hand (1995) subsequently extended the traditional biplot, where all the variables and observations could be displayed in a single plot - similar to the scatter plot for data containing two variables. In PCA biplots the variation between the variables is represented by Euclidean distances. Euclidean distances are defined as the distance between two points in Euclidean space as measured by a straight-line. The Euclidean distance is calculated using the Pythagorean formula for points in an n-dimensional space as (Anton and Rorres, 1994): $d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$. The interpretation of the observations and reading values on the biplot are therefore similar to that of a scatter plot in which points are also represented by Euclidean distances.

Now, using an example, the interpretation of biplots is discussed by extending a two-dimensional scatter plot to include the visualisation of more than two variables. The aim of this example is to assist the reader in understanding how to interpret biplots. When interpreting both scatter plots and biplots in this section, an important distinction is made between the interpolation of data points and the prediction of data points. Interpolation is defined as adding new points to a plot, and prediction is defined as reading off the values of the points in the plot. The measure to which one can use the plots to read off (predict) values of the observations is defined as the predictivity of the plot. The data considered in the example, is an arbitrary 4×4 data matrix, which contains 4 observations and 4 variables. The data matrix is tabulated in table 3.1.

Obs.	V1	V2	V3	V4
1	100	90	70	50
2	70	90	110	130
3	90	110	130	150
4	90	70	50	110

Table 3.1: Data matrix

A two-dimensional scatter plot is plotted to represent the first three observations of the first two variables, as plotted in figure 3.1. The values of the variables for each observation are read off the plot by drawing perpendicular lines from the observations to the axes. In two-dimensions it can be seen that the observations are represented accurately in the scatter plot. Additionally, new points can easily be interpolated on the graph. A new point is added where perpendicular lines drawn from the V1 and V2 axes intersect.

Next, in figure 3.2, the axes of the scatter plot in figure 3.1 are inversed and shifted. The horizontal axis representing V1 has been inversed and shifted upwards, and the vertical axis representing V2

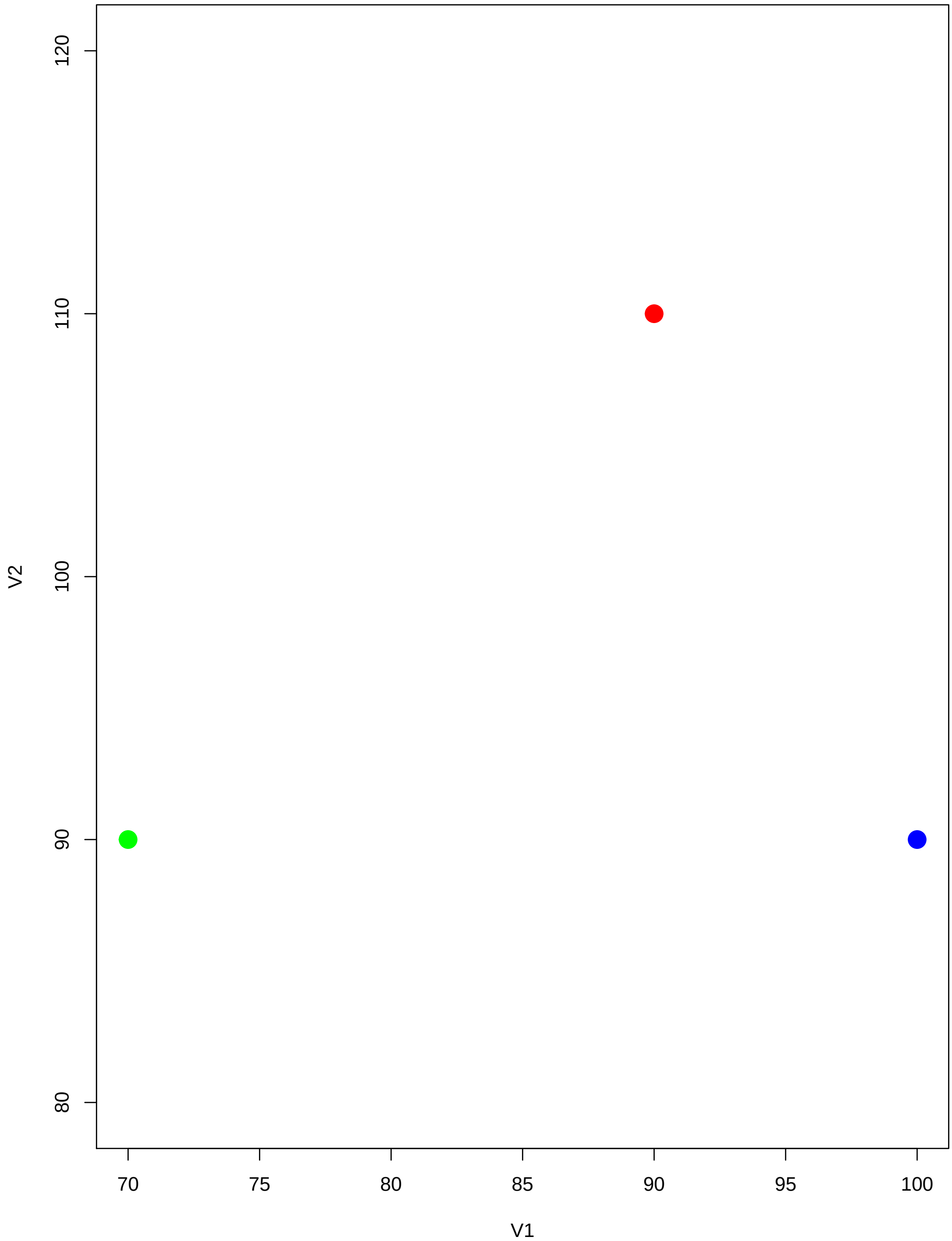


Figure 3.1: Scatter plot of the first three observations of the first two variables, V1 and V2

has been shifted to the right. It can be seen that as a result of the inversion of the V2 axis, the observations in figure 3.2 are a mirror image of those in figure 3.1. Although the axes of the scatter plot have been shifted and inversed, the values of the variables can still be read off from the plot, with no loss of accuracy. This is seen in figure 3.2, where the perpendicular lines are used to read off the values of the observations for each variable.

Now, in figure 3.3, a scatter plot in which the axes from figure 3.2 are rotated is plotted. One can see that, although the plot no longer looks like traditional scatter plot, the interpretation thereof remains the same. Additionally, adding new points to the plot remain the same. These examples illustrate that inverting, shifting, or rotating axes in a two-variable scatter plot do not influence the interpretation and accuracy of the plot's representation of the variables and observations.

Next, a three-dimensional graph of the first three observations of the first three variables is plotted in figure 3.4. The three-dimensional plot provides an accurate representation of the observations, with no loss of accuracy in reading off the values of the variables in the three-dimensional plot. Reading off the values in the three-dimensional plot are, however, not as simple as in the two-dimensional case. In the three-dimensional plots, two sets of perpendicular lines need to be drawn to read off the values in the plot: one set towards the variable plane, and one set upwards towards the axes' values. New points can also easily be added to the three-dimensional plot. New points are added where perpendicular lines drawn from the values of the three variables for the new observations intersect.

Now, in figure 3.5, the previously plotted three-dimensional plot is collapsed to represent the observations in a two-dimensional space. This plot of the three-variables is a biplot representation of the first three observations of the first three variables of the data matrix. Although the plot in figure 3.5 is drawn in one fewer dimension than in figure 3.4, there is no loss of accuracy in reading off the values of the variables in the two-dimensional biplot representation. The values of the observations can be read off in figure 3.5 similarly to that of the two-dimensional scatter plot through drawing perpendicular lines to the variable axes to read off the values. Interpolating new values in a biplot is much more complicated than for the two-dimensional scatter plots. Plotting of the new values requires more mathematical calculations to accurately represent the observation in the biplot space, and will not be discussed further. The derived mathematical calculations can

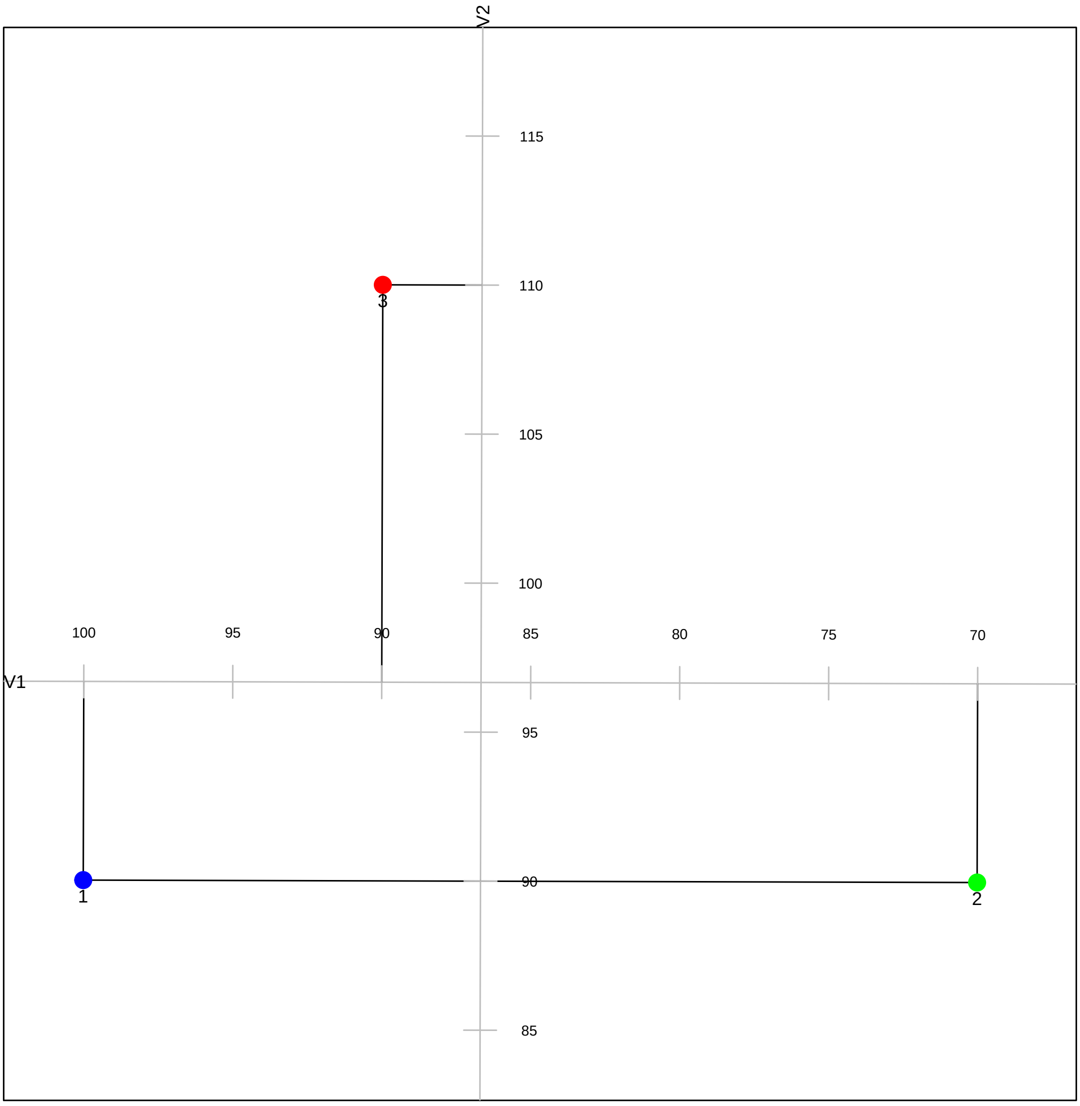


Figure 3.2: Scatter plot of the first three observations of the first two variables, V1 and V2, with shifted and inversed axes

Scatter plot of the first three observations of the first two variables, V1 and V2

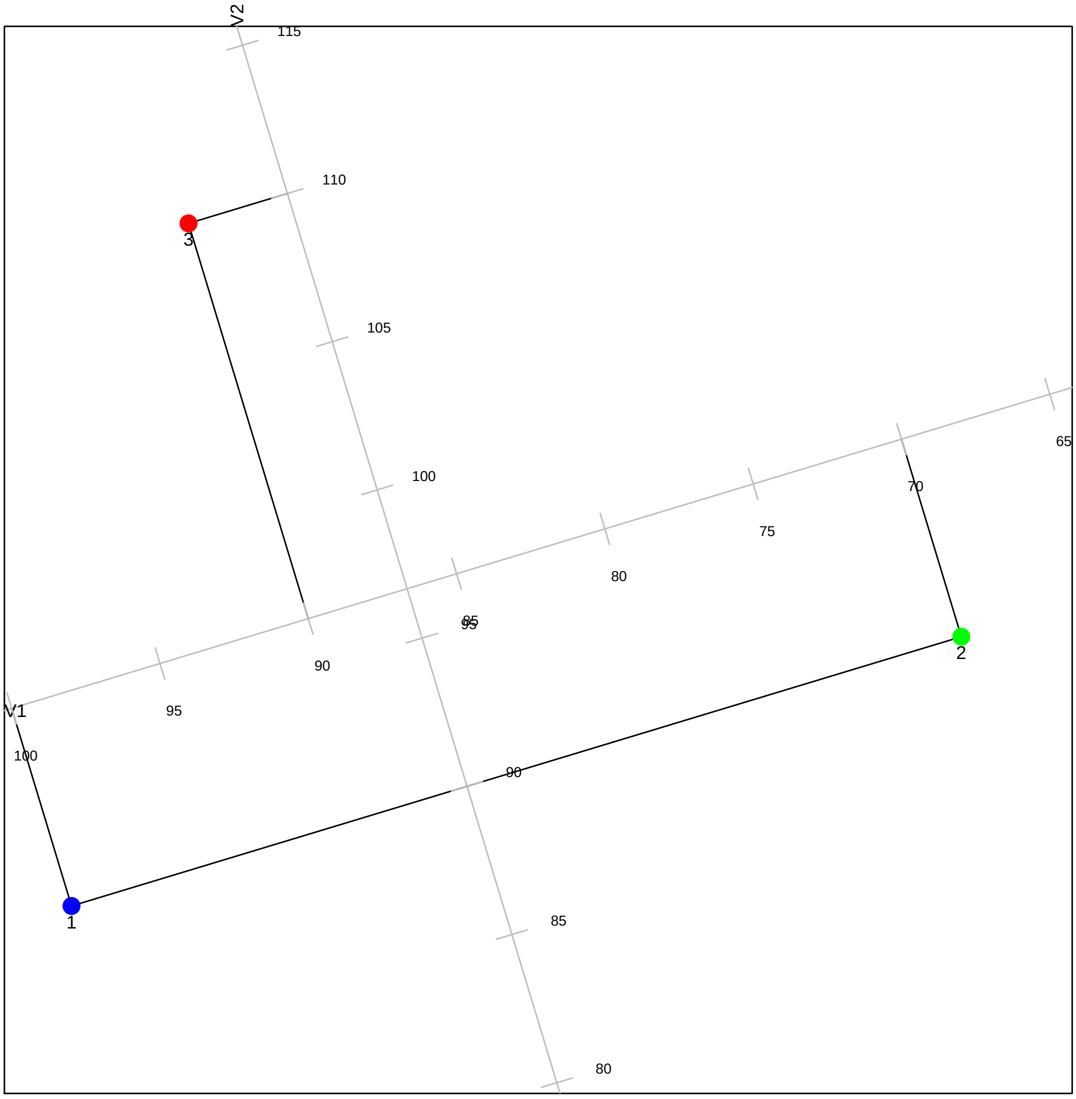


Figure 3.3: Scatter plot of the first three observations of the first two variables, V1 and V2, with rotated, shifted and inversed axes

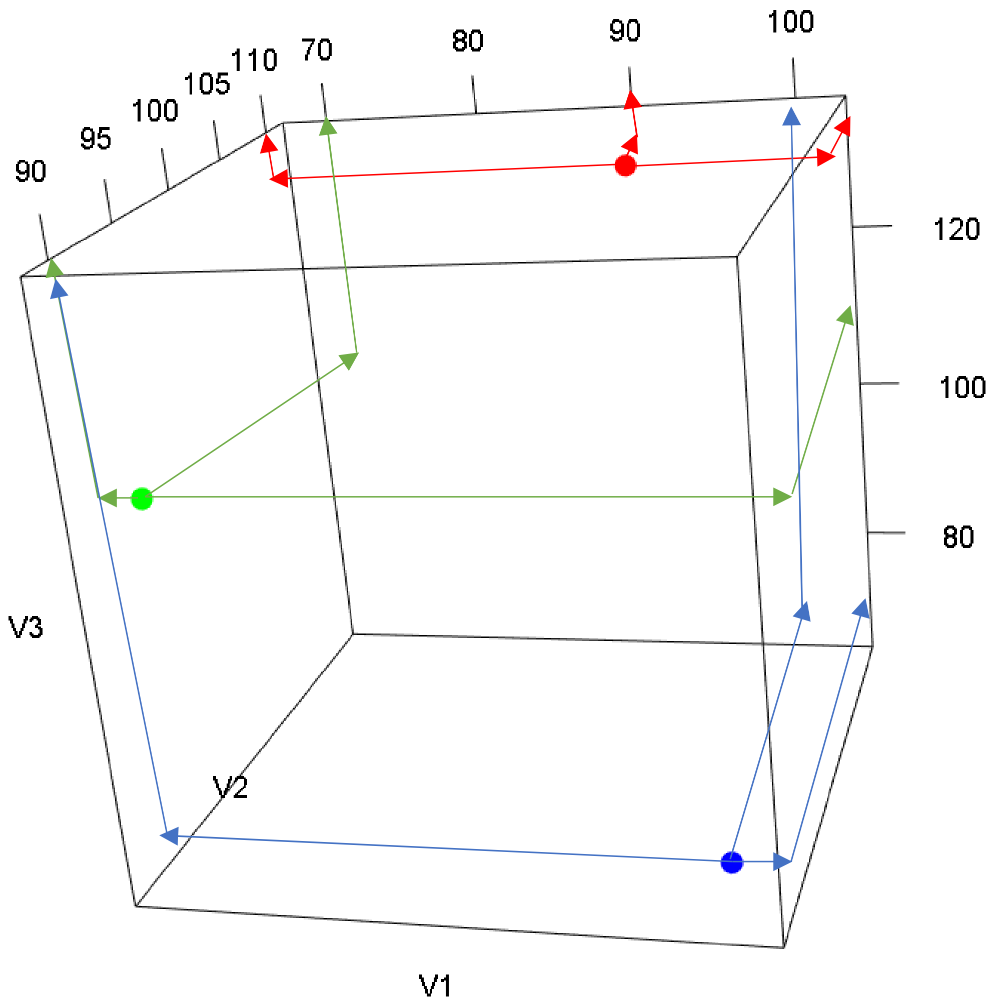


Figure 3.4: 3D Plot of the first three observations of the first three variables, V1, V2, and V3

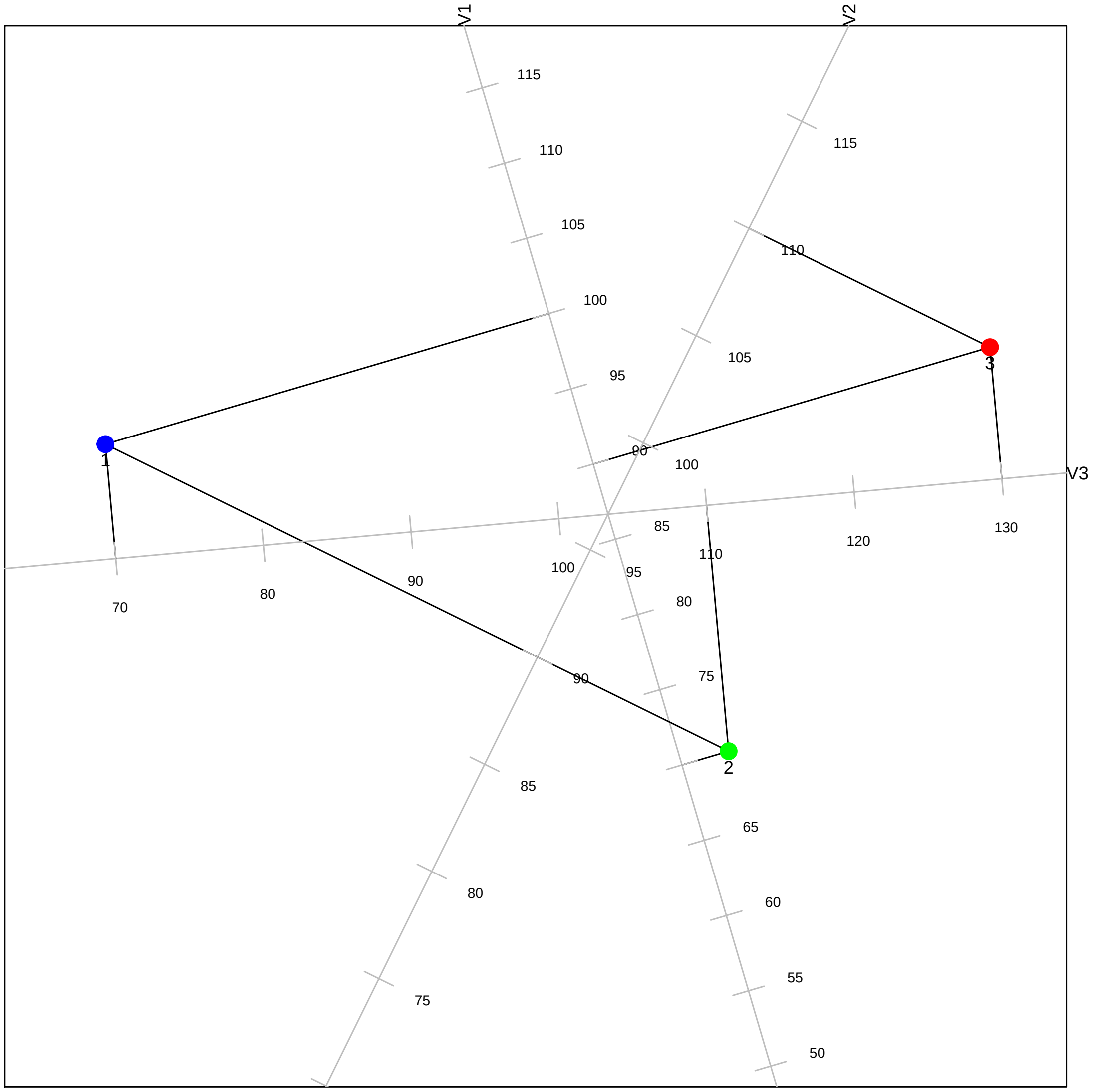


Figure 3.5: Biplot of the first three observations of the first three variables, V1, V2 and V3

be read in more detail in Gower *et al.* (2011) and Greenacre (2010).

Next, a fourth variable is added, and the corresponding biplot of the first three observations of all four variables of is plotted in figure 3.6. The interpretation of the biplot remains the same as in the previous biplot in figure 3.5. The values for all four variables can be read off from the biplot by drawing perpendicular lines from the observations to the variable axes and recording their corresponding values. The representation of the three observations in the biplot space with four-variables is accurate, where the values of the observations can be read off the plot without error.

Lastly, in figure 3.7, a biplot which represents the full data matrix in table 3.1 is plotted. In figure 3.6, the values of the three observations in the biplot space with four variables, can all be read accurately from the biplot without any loss of information. While in figure 3.7, however, the values of the four observations in the biplot space with four variables cannot be read off from the plot with 100% accuracy. There will be an error (i.e. loss of information) in representing observations in a biplot for $n > 3$ observations, as a plane cannot be drawn through all the points. This error can be seen in figure 3.7, where the values of the variables for the observations are not the same as their true values (table 3.1). The predicted values of the observations using the biplot are recorded in table 3.2.

Obs.	V1	V2	V3	V4
1	99.7	89.9	70.1	49.9
2	83.0	95.0	106.7	133.7
3	81.1	106.2	132.2	147.5
4	86.1	68.5	51.0	108.9

Table 3.2: Data matrix predictions

The discussed example illustrates how the interpretation of biplots can be done by extending the two-dimensional scatter plot to include more variables. It was also seen that when representing a data matrix with more than three observations in a biplot, the accuracy of the plot needs to be considered, where errors in predicting values using the biplot occur.

The *UBbipl* (le Roux and Lubbe, 2013) package in R, was used to draw the plots in this section.

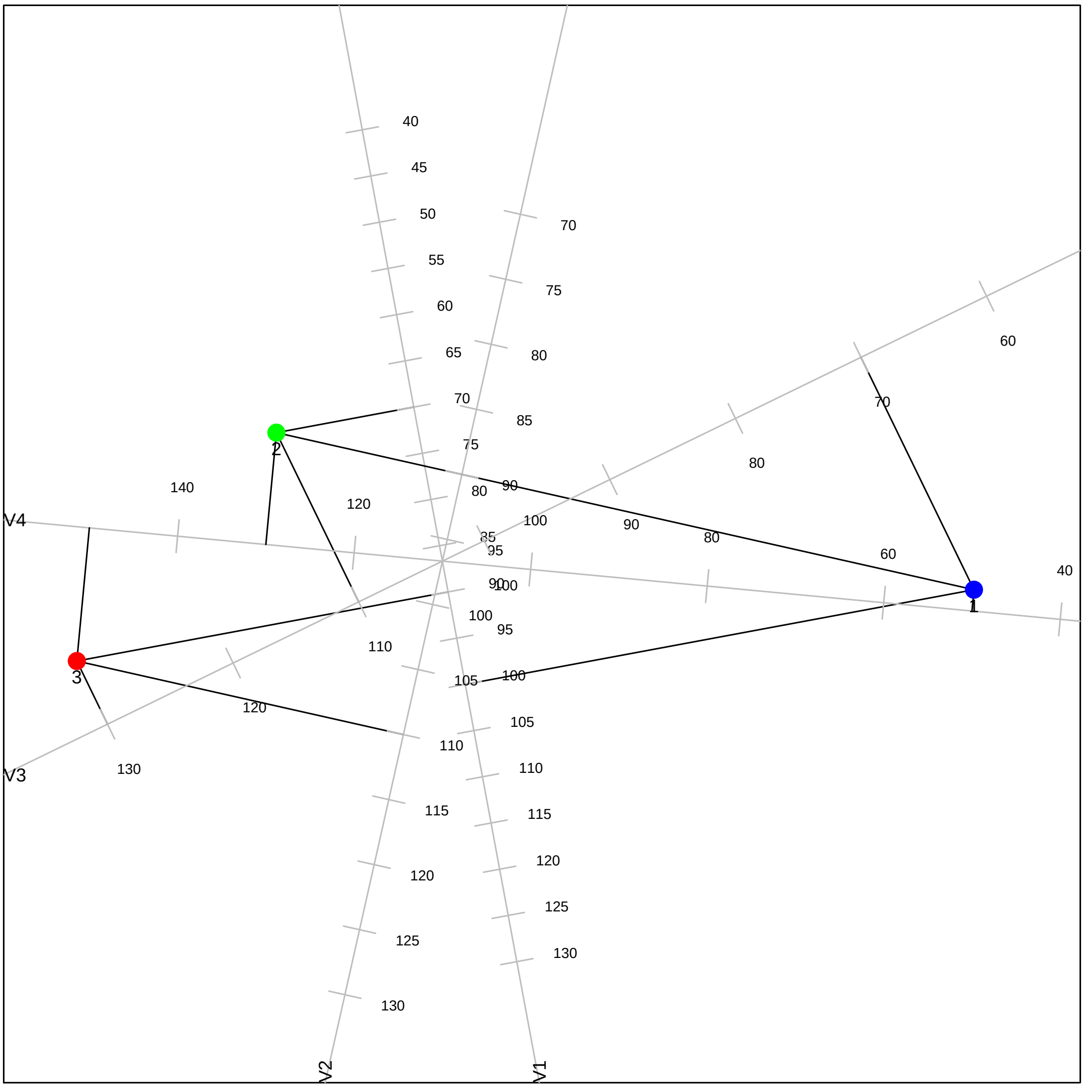


Figure 3.6: Biplot of the first three observations of the four variables, V1, V2, V3, and V4

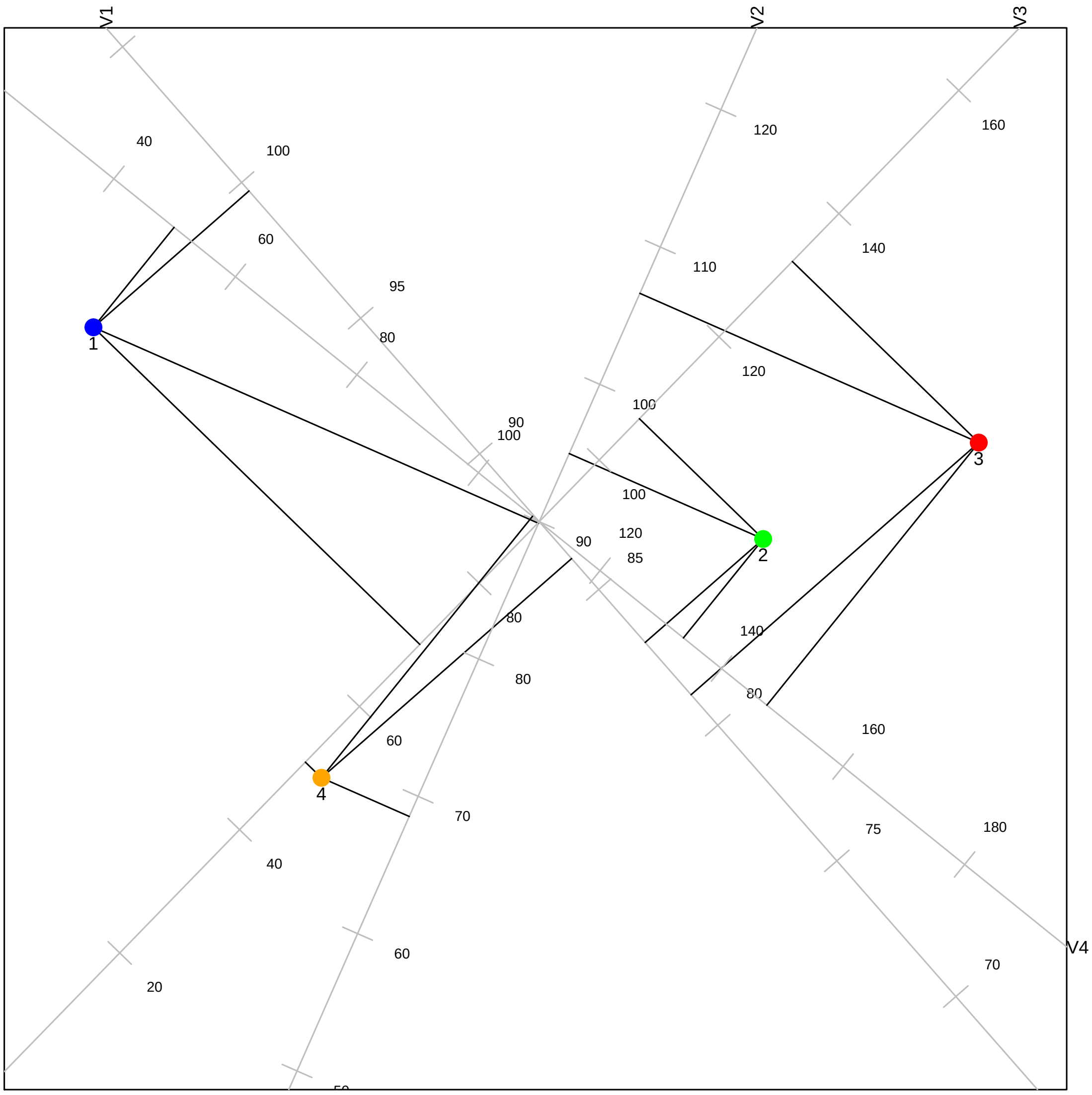


Figure 3.7: Biplot of the full data matrix

In the following section, the mathematical derivations used to construct biplots are discussed.

3.2 PCA BIPLOTS THEORY

As discussed in the previous section, biplots can be used to represent a data matrix, $X : n \times p$, in a r_{dim}-dimensional plot. In this study, the data matrix is assumed to be column-centered. Further, the biplot representations considered will all be in two-dimensions, i.e. $r_{dim} = 2$.

PCA will used to approximate the data matrix, X, in r_{dim} dimensions, denoted by the matrix $\hat{X}_{[r_{dim}=2]}$. PCA is a dimension reduction technique in which the approximated matrix in r_{dim} dimensions is calculated such that the maximum variation of the data matrix X in p dimensions is preserved. PCA is a technique which can be used to minimise the error between the data matrix X and the approximated matrix $\hat{X}$, as proved by the Eckart-Young theorem (Eckart and Young, 1936). The steps in calculating approximated data matrix $\hat{X}$ are discussed next.

The PCA method implements singular value decomposition of the data matrix X, and is denoted by:

$$X = U\Sigma V', \tag{3.1}$$

where U: an $n \times k$ orthonormal matrix, Σ: an $k \times k$ diagonal matrix, with the diagonal elements containing the nonzero singular values of X, and V: a $p \times p$ orthonormal matrix containing the eigenvectors of X.

The approximated data matrix is then derived as:

$$\hat{X}_{[r_{dim}=2]} = U\Sigma_{[r_{dim}=2]}V', \tag{3.2}$$

where $\Sigma_{[r_{dim}]}$ is a diagonal matrix, with the first $r_{dim} = 2$ diagonal elements containing the nonzero singular values of X and the remaining diagonal elements are zero.

The interpolation of a new observation on the biplot space, $\mathscr{L}$, is approximated as (Gower and Hand, 1995):

$$x'_{proj} = x'V_{r_{dim}}V'_{r_{dim}}. \tag{3.3}$$

The representation of the new observation in terms of the orthogonal biplot space, is approximated as (Gower and Hand, 1995):

$$z' = x'V_{r_{dim}}. \tag{3.4}$$

The z' values are the coordinates of the new observation in the biplot space, $\mathscr{L}$, and the first two columns of $V_{r_{dim}}$ are the scaffolding axes which form the basis of the biplot space.

The predicted values for an observation in X, can be approximated using the biplot as (Gower and Hand, 1995):

$$\hat{x}' = z'V'_{r_{dim}}, \tag{3.5}$$

where z' values are the coordinates of the observation in the biplot space, $\mathscr{L}$.

3.3 SUMMARY

PCA biplots form the basis of the biplot interpolation methodology which is discussed in the following chapter. The aim of this chapter was to provide the necessary theory required to interpret and construct biplots. To interpret biplots, a scatter plot example was discussed, and to construct biplots the necessary background theory was detailed. Further reading for more detailed mathematical derivations on constructing biplots, the reader is referred to Gower *et al.* (2011) and Greenacre (2010).

CHAPTER 4

METHODOLOGY

In this chapter, a biplot interpolation methodology is proposed that will be used to estimate the parameters for beta functions in order to approximate EE values. The methods used to calculate EE were discussed in detail in chapter 2. In chapter 3, the theory of PCA biplots was covered. Using the discussed theory as background, the biplot interpolation methodology is derived. The biplot interpolation method uses two data sets (training data), namely an input data set and an output data set, in order to interpolate output values for a new set of input data. This biplot interpolation method is then applied to IRS data, where an input grid of IRS data, and an output grid of estimated beta function parameters is used. The proposed biplot interpolation method is then used to estimate beta function parameters for a new set of IRS input parameters. These interpolated values for the beta function parameter estimates are then be used in estimating EE values. This biplot interpolation method aims to simplify the estimation of EE calculations. The chapter concludes with the testing methodology in which a standard deviation interval approach is proposed to see how well the estimated values using the biplot interpolation method compare to their true values within certain standard deviation bands.

In order to implement the biplot interpolation methodolody to EE data, the biplot interpolation is firstly discussed for a random set of training data in section 4.1. Thereafter, the application of the methodology to EE data is discussed, starting with defining an input grid in section 4.2, and thereafter creating the input grid in section 4.3. The input grid is created using various combinations of IRS parameters. The IRS input parameters used are interest rate curve data, swap rates, and maturities.

Next, using the created input grid of IRS data, the output grid of beta function parameter estimate values are calculated. In section 4.4, the steps used to calculate the output grid values are covered. The EE profiles are calculated for each entry of the input grid. Thereafter, the beta function parameters are estimated for each of the calculated EE profiles.

Once the input grids and output grids have been created, biplots are then drawn to link the two grids. Firstly, input data are used to draw biplots for various maturity ranges. The input data biplots along with the output grid data, are then used to estimate values of beta function parameters for new test input data. In section 4.5 the biplot interpolation methodology is extended to apply to the EE data.

Lastly, in section 4.6, the methodology is tested. The estimated beta function parameters are used to estimate the EE profile. The resulting profile is tested using a standard deviation band test. In figure 4.4, a diagram in which an overview of the steps used to implement the proposed biplot interpolation method is provided. This diagram can be used as a reference in this chapter in order to help the reader understand the sequence of the steps used in the proposed biplot interpolation method.

4.1 BIPLOT INTERPOLATION METHODOLOGY

In this section, the biplot interpolation method is explained in general, and in the subsequent sections the method will be used to apply to IRS data. The biplot interpolation method is used to estimate output parameter values for a given input parameter set. An input data set, X: an $n \times p_{input}$ matrix, and an output data set, Y: an $n \times p_{output}$ matrix, will be used in the biplot interpolation method. The steps in estimating output values, $\hat{Y}_{new}$: an $n_{new} \times p_{output}$ matrix, for a set of new input values, X_{new}: an $n_{new} \times p_{input}$ matrix, are outlined below:

1. A PCA biplot is drawn for the input data matrix and the resulting coordinates data is stored to be used in steps 2 and 4. The coordinates data are denoted with $(Z_{x,input}, Z_{y,input})$, and is calculated as $Z = XV_{[2]}$, where X is the centered input data set, and $V_{[2]}$ are the first two eigen vectors which form the basis for the biplot space.

2. A contour map overlaying the PCA biplot is plotted separately for each of the p_{output} output data set parameters. For a specific output variable (p_{output_i}: a $n \times 1$ column vector), the contour map is plotted using tri-linear interpolation. The tri-linear interpolation is used to link the output parameter values (p_{output_i}), to the coordinates data values (Z : an $n \times 2$ matrix). The contour map helps to visualise the link between the input variables and each of the output variables. The link between the coordinates data $(Z_{x,input}, Z_{y,input})$ and the output data for each output parameter can then be determined.

   The $(Z_{x,input}, Z_{y,input})$ denotes the coordinates of the input matrix in the biplot space. Therefore, the input matrix observations can be linked to the output parameter values which overlay the biplot. For a new input observation, the position of the new observation in the biplot space can be determined, i.e. the corresponding coordinates data. An estimate for the ouput parameter can then be interpolated by using the calculated coordinates data of the new observation and the overlaying contour map data.

3. A new input data set, X_{new}, which include values not contained in the input data set, X, is then used to estimate output parameter values, $\hat{Y}_{new}$. The coordinates of the new input values Z^{new}_{input} in the biplot space, are calculated as: $Z^{new}_{input} = X^*_{new}V_{[2]}$, where X^*_{new} contains the new input values centered using the original input mean values, and $V_{[2]}$ is as defined in step 1 which corresponds to the original input matrix X. The Z^{new}_{input} values will be used in the following step to estimate the values for each of the output variables.

4. A tri-linear interpolation is performed to estimate the values for each of the output variables. For the calculated Z^{new}_{input} values, the value for each output variable ($\hat{Y}_{output,new_i}$, for i = 1, ..., p_{output}) is interpolated consecutively. The tri-linear interpolation uses training data which include the original Z values, and the original output parameter values, Y, to estimate the new output parameter values.

The methodology can be implemented in R using the *UBbipl* package and the R function *interp*. The *UBbipl* package can be used to draw the biplots and calculate the Z^{new}_{input} values in step 3. The R function *interp* can be used to perform the tri-linear interpolation in step 4, and in plotting the contour plots in step 2. The *interp* function uses two different interpolation methods, namely Barycentric piecewise-linear interpolation and Akima splines interpolation. Both interpolation methods were implemented in this study. A limitation of the Barycentric piecewise-linear interpolation method is that it cannot extrapolate output values for test data observations which fall outside the convex hull of the input values provided. Therefore, the Akima splines interpolation method was also used, because output values could be estimated using extrapolation for this method.

The estimation of the output variables using the biplot methodology can be tested by comparing the estimated output values to the true values of the output variables. In the following sections, an application of the biplot interpolation method is considered for EE data of interest rate swaps. Firstly, the creation of the input grid used in the application is discussed.

4.2 GRID INPUTS

The EE values are calculated using the xVA package function: $CalcSimulatedExposure$. The $CalcSimulatedExposure$ function uses the Monte Carlo simulation approach to estimate EE values for the input swap parameters. The function uses the following inputs:

- $discount\_factors$: a $1 \times N$ array, with N: the number of cashflows. The $discount\_factor$ array contains the discount curve corresponding to the spot curve. For the application, the zero curves are first parameterised using the Nelson-Siegel model.

- $spot\_curve$: a $1 \times N$ array. The $spot\_curve$ array contains the curve with the market spot rates. For the application, the zero curves are first paramterised using the Nelson-Siegel model. The Nelson-Siegel fit curves are then used as a proxy for the true spot curves.

- $trades$: a list of trades, with each trade consisting of three inputs, namely E_i: maturity, pay_leg_rate: the fixed swap rate, and $BuySell$: the buy or sell position. The trades list contains the trade information of the input interest rate swap to be evaluated.

- $time\_points$: a $1 \times N$ array. The $time\_points$ array contains the time points for which the exposure values are calculated, which correspond to the cashflow payment dates.

- sim_data: a list of simulation parameters, with the following inputs, namely num_of_sims: the number of simulations, a: the mean reversion factor for the Hull-White Model, and σ: the volatility parameter for the Hull-White Model. The sim_data list contains the simulation data which is used to simulate the interest rate, using the Hull White model.

A grid of EE values is calculated for a defined set of grid inputs. To set up the grid, the following parameters will be chosen to be fixed: the notional value, *Not*, is set equal to 1, and the payment frequency, pmt_{freq}, is chosen to be fixed as three months. The Hull-White model parameters for the mean reversion rate and volatility, are chosen arbitrarily as $a = 0.1$, and $\sigma = 0.2$. The estimation of the parameters for the Hull-White model are beyond the scope of this study.

There will be five parameters that will be varied in the grid. These include the maturities, the fixed rates, and the spot curve inputs. The maturities are chosen as $T \in \{1,\ 5,\ 10,\ 15,\ 20,\ 25,\ 30\}$ and the fixed rates are chosen as $r_{fix} \in \{0.03,\ 0.04,\ 0.05, ...,\ 0{,}14\}$. The spot curve inputs include three parameters, L, S, and C, which correspond to the Nelson-Siegel model. The chosen values for the three Nelson-Siegel parameters are discussed in the following subsection.

4.2.1 Nelson-Siegel input parameters

In this section a range of possible values for the Nelson-Siegel parameters, which will be used to create the input grid, are determined. The range of possible values for the parameters were determined using historical ZAR zero yield curve data. The data used includes 2867 daily observations of South African (ZAR) zero yield curve data for the time between 2005/06/09 and 2016/10/31.

4.2.1.1 Parameter estimates

The R package *YieldCurve* (Consiglio and Guirreri, 2011) is used to estimate Nelson-Siegel parameter values for each of the 2867 daily ZAR zero curve data. The Nelson-Siegel parameter estimate function is adjusted to calculate the parameter estimates L, S, and C for a fixed value of the λ parameter. According to Diebold and Li (2006), for data measured in terms of year fractions, a fixed choice of $\lambda = 0.7308$ is proposed to obtain stable parameter estimate values.

Using the historical ZAR yield curve data, a large set of daily parameter estimates are obtained, consisting of three parameter estimates ($\hat{L}$, $\hat{S}$, and $\hat{C}$) for each observation. From the large set of daily parameter estimates, a range of parameter estimate values for each parameter is determined.

In determining the range of possible input values for the parameters, outliers for the parameter estimates must be taken into account as they could lead to unreasonable results, and therefore need to be removed. Outlier estimates correspond to observations of the ZAR zero curve data which occur rarely and therefore do not need to be included in the input values used when setting up the input grid. A box plot is used to detect and remove the outliers from the estimated parameters. All parameter estimates which do not fall within the box plot boundary points are considered to be outliers and are consequently removed from the data set. The range of parameter estimate values contains the values which are bounded by box plot boundaries. The lower boundary value of the box plot is defined as $Q_1 - 1.5(IQR) = Q_1 - 1.5(Q_3 - Q_1)$, and the upper boundary value is defined as $Q_3 + 1.5(IQR) = Q_3 + 1.5(Q_3 - Q_1)$, where Q is defined as the quartile and IQR is defined as the inter quartile range.

In table 4.1 the boundary points and the number of outliers for each of the NS parameters, calculated using the ZAR data set, are given.

	$\hat{L}$	$\hat{S}$	$\hat{C}$
(Lower, Upper)	$(0.0663, 0.0967)$	$(-0.0662, 0.0327)$	$(-0.0748, 0.0712)$
Number of outliers	64	219	11

Table 4.1: The box plot boundary values and the number of outliers for each Nelson-Siegel parameter estimate fitted to daily ZAR data from 2005/06/09 to 2016/10/31

In figure 4.1, a box plot of each parameter is plotted, and in figure 4.2, a time-series plot of the estimated Nelson-Siegel parameters are plotted for each parameter.

4.2.1.2 Parameter estimate grid inputs

In the previous section, outliers in the calculated Nelson-Siegel parameter estimates were identified and removed using a box plot methodology. Boundary intervals were calculated for each parameter in which the identified outliers were omitted. The Nelson-Siegel parameter estimates which will be used to set up the input grid were chosen as the percentiles (P = 0.1, 0.2, ..., 0.9, where P is defined as the $P'th$ percentile) of the calculated boundary intervals, recorded in table 4.1. In table 4.2, the calculated percentiles of the Nelson-Siegel grid inputs are recorded.

Percentile	Parameters		
P	$\hat{L}$	$\hat{S}$	$\hat{C}$
0.1	0.0745	-0.0353	-0.0386
0.2	0.0765	-0.0305	-0.0255
0.3	0.0787	-0.0285	-0.0171
0.4	0.0801	-0.0264	-0.012
0.5	0.0813	-0.0202	-0.0038
0.6	0.0825	-0.0138	0.004
0.7	0.0839	-0.0099	0.0106
0.8	0.0864	-0.0052	0.0251
0.9	0.0901	0.0181	0.0338

Table 4.2: The Nelson-Siegel grid inputs

In section 4.2, all of the inputs used in the *CalcSimulatedExposure* function were defined. The

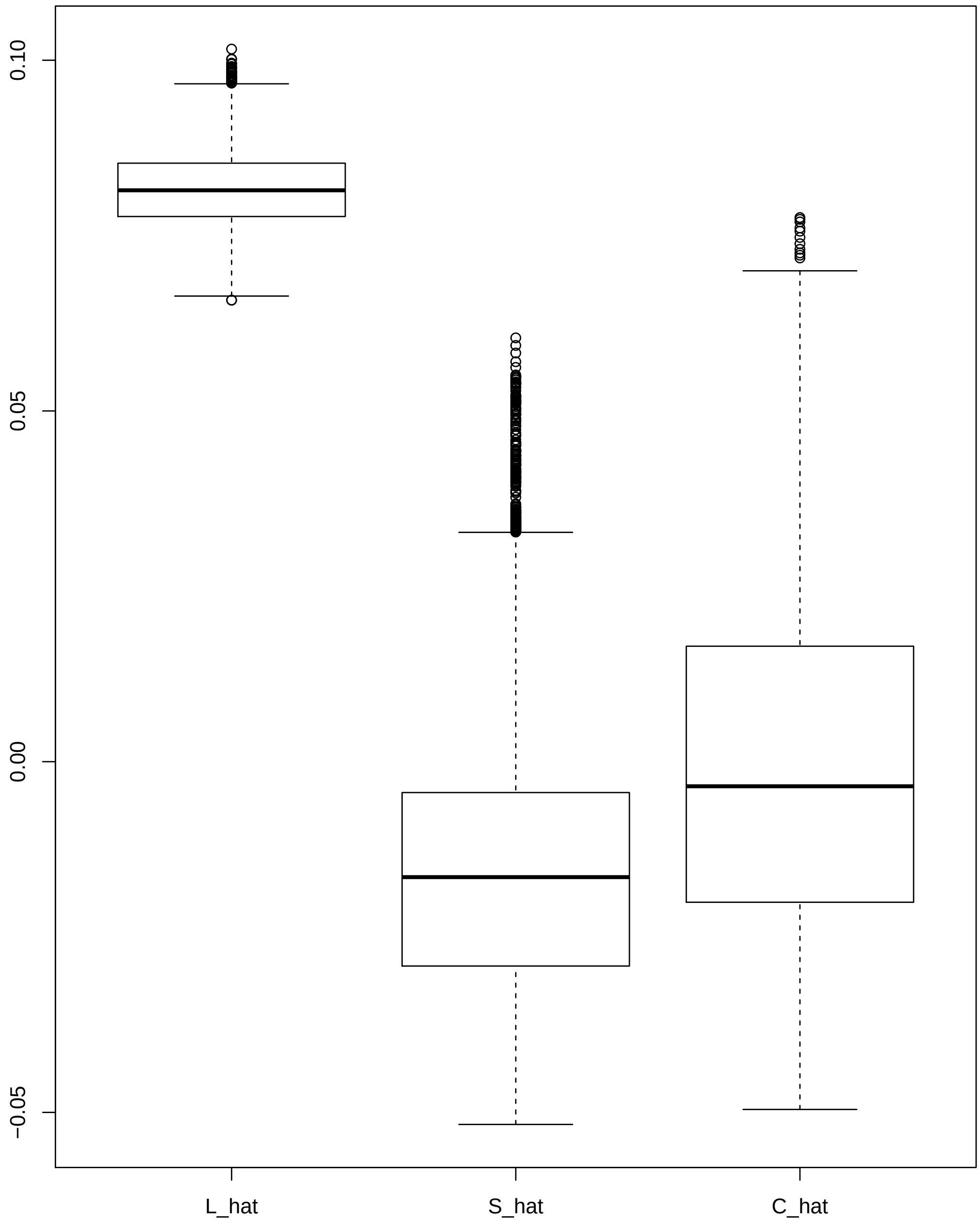


Figure 4.1: Box plot of ZAR Nelson-Siegel parameter estimates

A box plot for each Nelson-Siegel parameter estimate ($\hat{L}$, $\hat{S}$, and $\hat{C}$) is drawn. It can be seen that the level parameter estimates ($\hat{L}$) are not as widely spread in value, as those of the slope ($\hat{S}$) and curvature ($\hat{C}$) parameter estimates. For each of the parameter estimates, it can be seen that the majority of the outliers fall on the upper tail of the distribution.

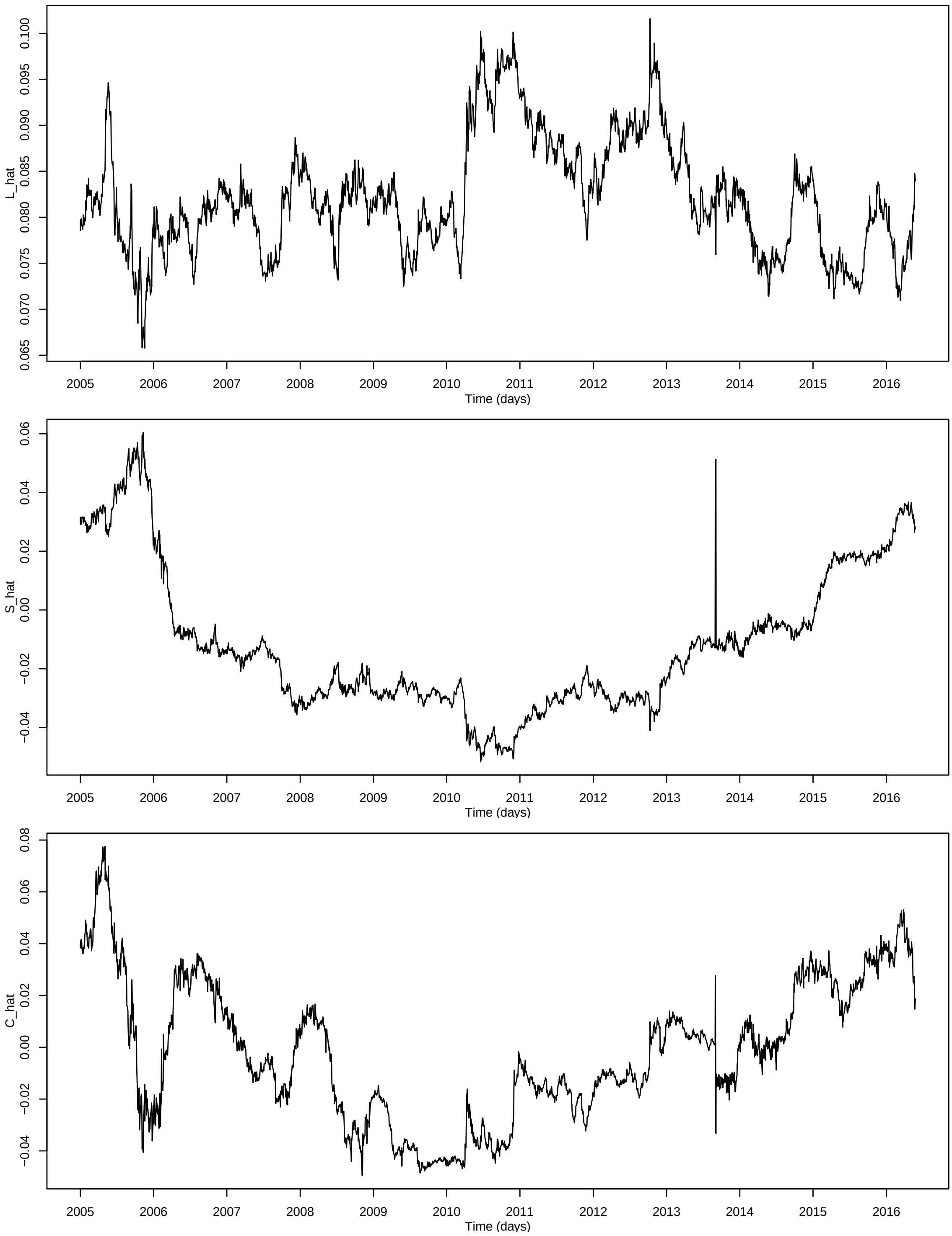


Figure 4.2: Nelson-Siegel parameter estimates plotted over time

Estimated values for each of the Nelson-Siegel parameters (L, S and C) are plotted. The estimated values were calculated using daily ZAR zero yield curve data for the time period 2006/06/09 to 2016/10/31. A large jump in the parameter estimate values is seen at time 2013/09, which corresponds to an outlier estimate for the parameters.

values used for the inputs were also discussed in this section. In the following section, the creation of the grid using the defined input parameters will be discussed.

4.3 CREATE INPUT GRID USING *XVA* PACKAGE

The input grid is set up to include all the possible combinations of the variable inputs, which include the Nelson-Siegel parameters ($\hat{L}$, $\hat{S}$, and $\hat{C}$), maturities (T), and swap rates (r_{fix}). The total number of possible input combinations for the variable parameters $\hat{L}$, $\hat{S}$, $\hat{C}$, T, and r_{fix} are equal to 61,236 $= 9 \times 9 \times 9 \times 7 \times 12$. The first three and last three data inputs of the grid are shown in table 4.3.

	$\hat{L}$	$\hat{S}$	$\hat{C}$	T	r_{fix}
1	0.0745	-0.0353	-0.0386	1	0.03
2	0.0765	-0.0353	-0.0386	1	0.03
3	0.0787	-0.0353	-0.0386	1	0.03
...	...	...	...	...	...
61234	0.0839	0.0181	0.0338	30	0.14
61235	0.0864	0.0181	0.0338	30	0.14
61236	0.0901	0.0181	0.0338	30	0.14

Table 4.3: An excerpt of the grid input

The EE is estimated for each entry of the input grid. The *xVA* package in R will be used to simulate the EE values for the swap grid inputs. The *xVA* function *CalcSimulatedExposure* discussed in section 4.2 was adjusted to only include the variable input parameters for the curve parameters, maturity and swap rate. In the following section, the calculation of the output grid parameters is discussed.

4.4 GRID OUTPUT

In the previous section the EE profile was estimated for each of the input grid parameters. For each calculated EE profile, the corresponding beta function parameters are estimated. The first three and last three data inputs in the grid are shown in table 4.4.

The creation of the input grid and the estimation of the output grid were discussed in sections 4.2

to 4.4. Using these input and output data sets, the biplot methodolody will be implemented to derive the link between these two data sets. The biplot methodology for the IRS data discussed is detailed in the following section.

	$A\hat{U}C$	$\hat{\alpha}$	$\hat{\beta}$
1	136.1076	4.3281	3.2284
2	96.6355	4.1560	3.1479
3	73.4922	3.9992	3.0896
...	...	...	...
61234	1.0417	1.1182	1.2176
61235	1.0391	1.1166	1.2168
61236	1.0362	1.1145	1.2160

Table 4.4: An excerpt of the grid output

4.5 EXPECTED EXPOSURE ESTIMATION

In testing the biplot interpolation methodology, input test data will be taken for both within-grid data and historical NS parameter data. Within-grid data is simulated for values which lie within the input grid boundary values, for each of the input parameters (L, S, C, r_{fix}, and T). It should be noted that the within-grid values are not necessarily points on the grid, but within the grid boundary values. Historical NS parameter data are simulated for values which include historical curve parameter estimates (L, S, and C), and within grid boundary values for the rate and maturity parameters (r_{fix} and T). Similarly, it is noted that within grid boundary values for the rate and maturity parameters (r_{fix} and T) are not necessarily points on the grid, but values that fall within the grid boundary values.

The test data contains interest rate swap data for maturities which fall within the grid maturity values. For a maturity which falls within two input grid maturities, e.g. 3 years, 12.5 years etc., the EE profile is estimated using a weighted average of two swaps.

The weighted average estimated exposure profile is calculated as follows:

1. Consider a t year swap, where t falls between T_i and T_j, for $i < j$.

2. The t year swap is characterised by the test input grid parameters L, S, C, and r_{fix}.

3. Given the test input grid parameters (L, S, C, and r_{fix}), the beta function parameters (AUC, α, and β) are estimated for both T_i and T_j, for $i < j$ maturities.

4. A t year profile is estimated as the weighted average of the T_i and T_j year profiles estimated with the parameters estimated in the previous step. The weighted average EE profile, $EE_{est}(t)$, for a t year interest rate swap is calculated as:

$$EE_{est}(t) = \left[1 - \frac{t - T_i}{T_j - T_i}\right] EE(T_i) + \left[\frac{t - T_i}{T_j - T_i}\right] EE(T_j) \tag{4.1}$$

   where, $EE(T_i)$ is the EE profile for a T_i year swap, and $EE(T_j)$ is the EE profile for a T_j year swap.

5. The estimated t year profile is compared to the true estimated EE profile.

In figure 4.3, the weighted average estimated EE profile for a 7.5 year swap is plotted. In the following section, the testing of the biplot methodology using EE data will be discussed.

4.6 TESTING OF THE BIPLOT INTERPOLATION METHODOLOGY

In testing the biplot interpolation methodology, test input data values are randomly chosen for both within-grid data as well as historical NS parameter data. A sample of $n_{test} = 100$ within-grid data and historical NS parameter data were chosen at random. The resulting output parameter estimates were used to estimate the EE curve. To test whether the biplot interpolation estimate provides an accurate estimation of the true EE curve, a standard deviation test was set up as follows:

1. Draw input test data. $X_{test,input}$ an $n_{test} \times p_{input}$ matrix, with n_{test} observations and p_{input} variables.

2. Using the input test data, run the biplot interpolation method as detailed in section 4.1 for each entry to obtain estimated EE curves (EE_{est}) for each of the input test data values.

3. For each of the input test data values simulate the true EE curve (EE_{TRUE}).

4. The standard deviation of the simulations of the exposure values are also calculated for each of the test input data values, and denoted as $SD(Exposure\,Simulations)$.

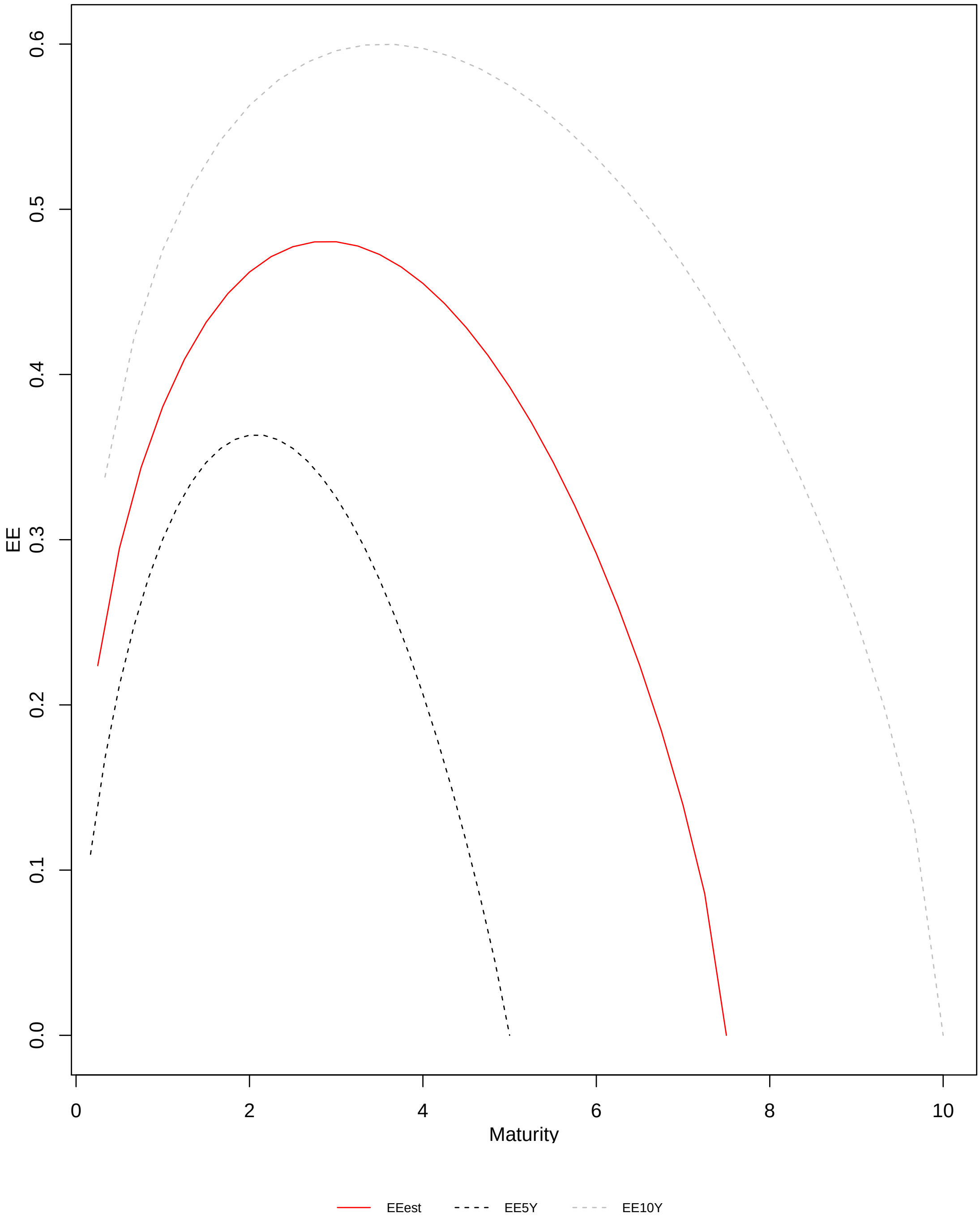


Figure 4.3: t Year Expected Exposure Plot

The $EEest$ profile is the EE profile plotted for a 7.5Y interest rate swap with parameter values: $L = 0.0864$, $S = -0.01379$, $C = 0.0040$, and $r_{fix} = 0.08$. The $EEest$ profile is calculated as the weighted average of the 10 year EE profile ($EE10Y$), and the 5 year EE ($EE5Y$).

5. An interval is set up as follows: $(upper, lower)_c := EE_{TRUE} \pm c \times SD(Exposure\, Simulations)$, where EE_{TRUE} is the true EE values, c is a constant, and $SD(Exposure\, Simulations)$ is the standard deviation of the exposure simulations.

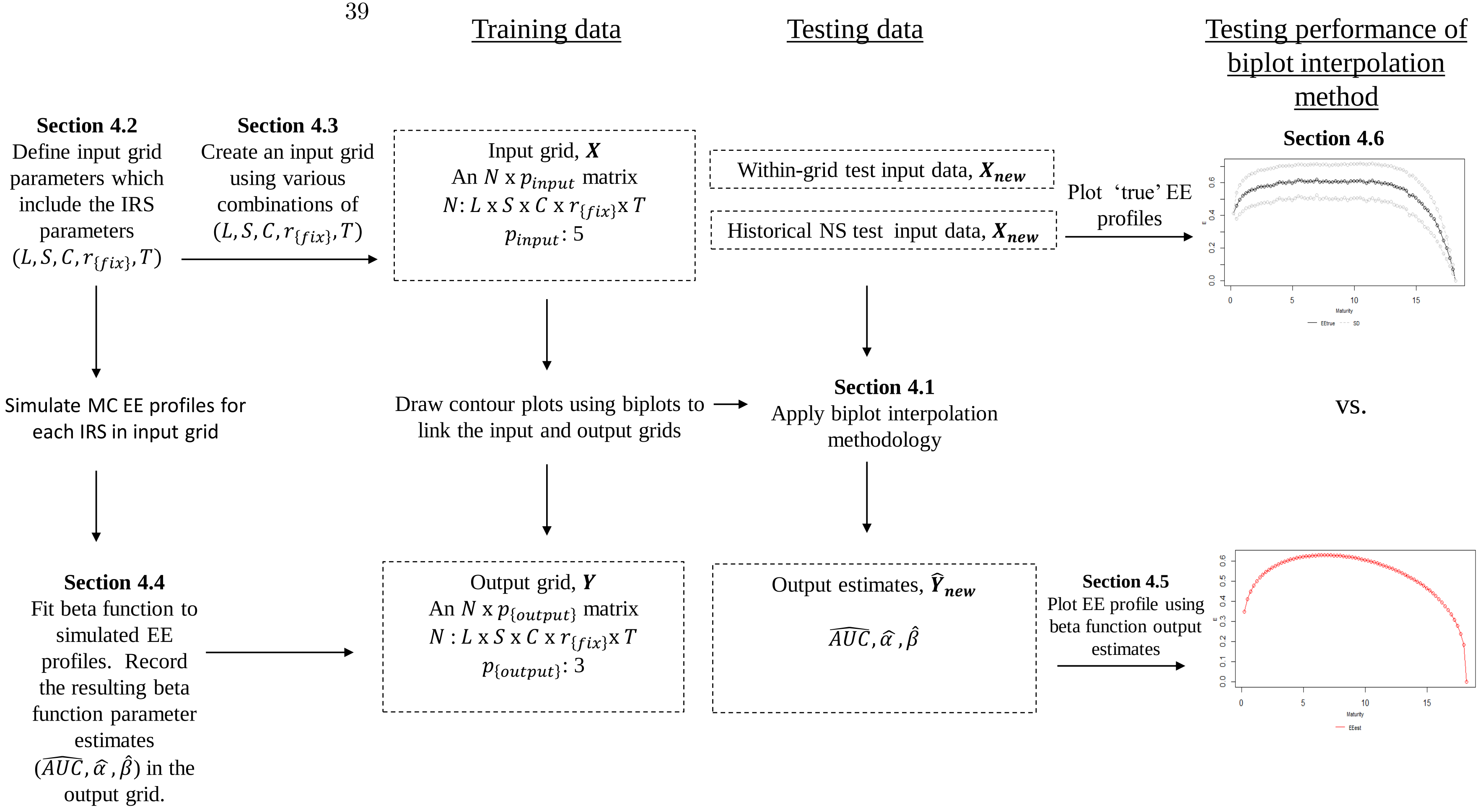
6. The error value is calculated as the percentage of the estimated EE (EE_{est}) values which do not fall within the interval calculated in the previous step. For test input i, where $i =$ 1, ..., n_{test}:
$$Error_{c,i} = \frac{1}{n_{EE}} \sum_{j=1}^{n_{EE}} \mathbb{1}_A\{EE_{est_{i,j}}\} \tag{4.2}$$
where, c is a constant representing the number of standard deviations, n_{EE} is the number of EE observations, $\mathbb{1}_A\{EE_{est_{i,j}}\}$ is the indicator function. The indicator function is defined for $EE_{est_{i,j}} \in A$, where A is the interval $(upper, lower)_c$.

7. The average error over the simulations for each maturity range is computed, and for varying values of $c = 0.5$, 1, 1.5, i.e. the number of standard deviations which determines the width of the interval.

The average errors are studied for varying standard deviation bands and at varying maturity ranges. The results are used to determine whether the estimated exposure curve is a reasonable approximation of the true EE values.

4.7 SUMMARY

The biplot interpolation methodology and its application to EE data was detailed in this chapter. The testing method of the biplot interpolation method, where a standard deviation interval approach was used, was also covered. A overview of the biplot interpolation methodology discussed in this chapter is seen in figure 4.4. The results from the biplot interpolation methodology for the input test data discussed in section 4.6, are provided in the following chapter.


Training data
Testing data
Testing performance of biplot interpolation method
Section 4.2
Define input grid parameters which include the IRS parameters $(L, S, C, r_{\{fix\}}, T)$
Section 4.3
Create an input grid using various combinations of $(L, S, C, r_{\{fix\}}, T)$
Input grid, $\boldsymbol{X}$
An N x p_{input} matrix
N: L x S x C x $r_{\{fix\}}$ x T
p_{input}: 5
Within-grid test input data, $\boldsymbol{X_{new}}$
Historical NS test input data, $\boldsymbol{X_{new}}$
Plot 'true' EE profiles
Section 4.6
Simulate MC EE profiles for each IRS in input grid
Draw contour plots using biplots to link the input and output grids
Section 4.1
Apply biplot interpolation methodology
vs.
Section 4.4
Fit beta function to simulated EE profiles. Record the resulting beta function parameter estimates $(\widehat{AUC}, \hat{\alpha}, \hat{\beta})$ in the output grid.
Output grid, $\boldsymbol{Y}$
An N x $p_{\{output\}}$ matrix
N : L x S x C x $r_{\{fix\}}$ x T
$p_{\{output\}}$: 3
Output estimates, $\boldsymbol{\hat{Y}_{new}}$
$\widehat{AUC}, \hat{\alpha}, \hat{\beta}$
Section 4.5
Plot EE profile using beta function output estimates

CHAPTER 5

RESULTS

In this chapter randomly chosen test input data is used to test the biplot interpolation methodology which was discussed in detail in chapter 4. This chapter begins with a data analysis of the training grids, which includes the input grid values and the output grid values that will be discussed in section 5.1. The input grid values contain IRS parameters and the output grid values contain the estimated beta function parameters. Two sets of test input data will be used to test the biplot interpolation method, and the resulting graphs and error results are provided for each of these sets of data in sections 5.2 to 5.3.

5.1 DATA ANALYSIS OF THE TRAINING GRIDS

The data used in setting up the interpolation study is discussed in further detail. Biplots and contour maps are drawn to visually analyse the data. Firstly, the biplot of the input grid parameters (L, S, C, and r_{fix}) is drawn for varying maturities, after which contour maps are drawn for each of the beta function output parameter variables, namely AUC, α, and β.

5.1.1 Biplots

The input grid is split for various maturity ranges, this split has been chosen as the time to maturity has the most significant impact on the estimated values in the output grid. As an example, the input grid for 15 year maturity swap data is plotted in figure 5.1. The input grid biplots for the other maturity ranges will result in similar plots. These biplots are used in the interpolation of the test input data. From the biplot, the range of input parameter values and the dispersion of the input grid parameter values can be seen. The biplot provides an overview of all the values of the variables of the input data grid. When interpolating new test input values, the biplot representation can be used to visualise where the new test data value lies in relation to the grid inputs.

5.1.2 Contour Maps

The input biplot data as well as the output grid data for each parameter is used to interpolate output parameter values for input test set swap values. The output grid data contains columns

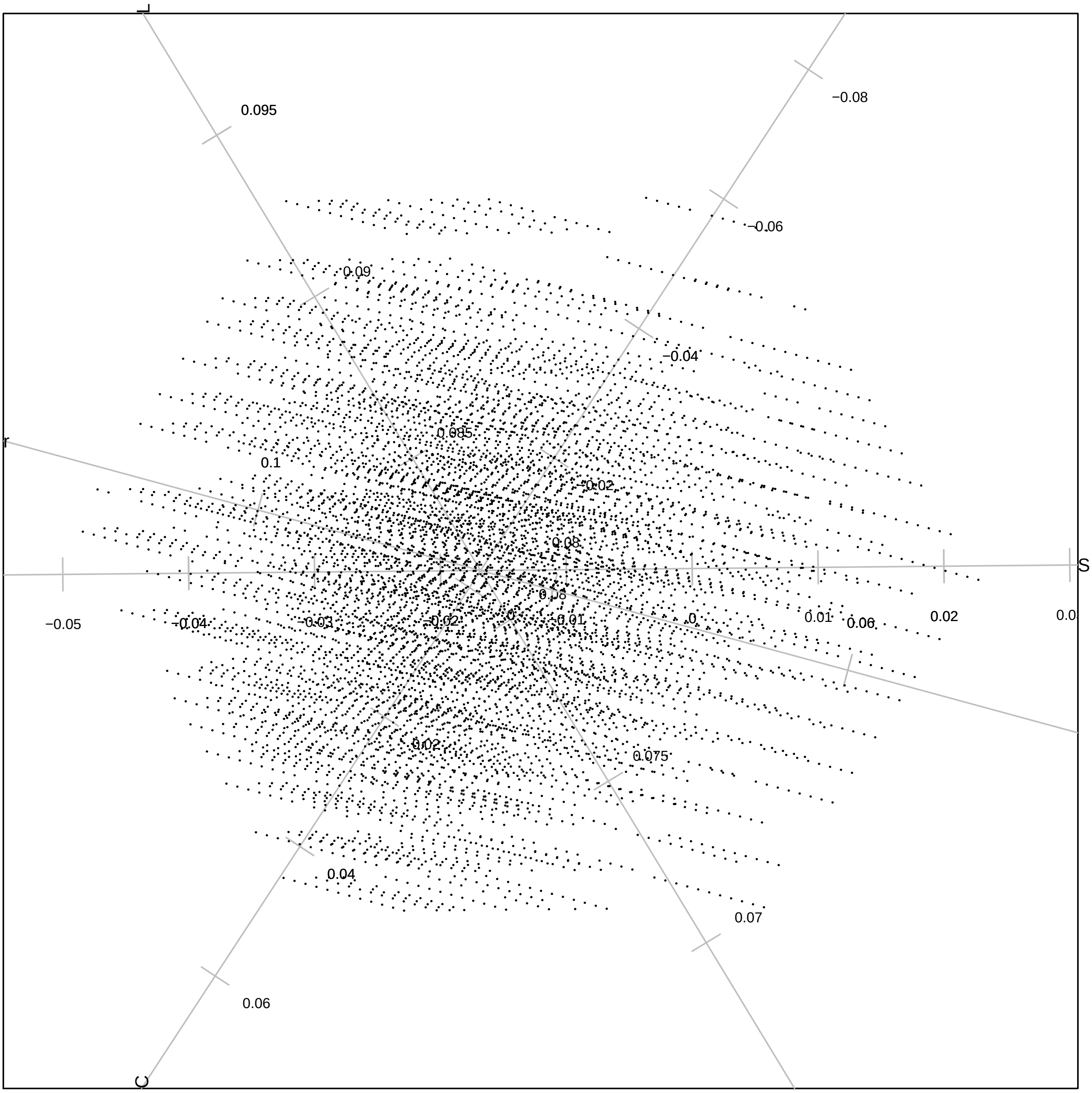


Figure 5.1: Input biplot

A biplot of the input grid is plotted. The input grid parameters (L, S, C, and r_{fix}) define the input swap parameters for the Nelson-Siegel curve parameters, and swap rate. The biplot is a useful tool to visualise the range of values for each of the input parameter values which are represented by the axes in the plot, on a single plot. The input biplot is further used to interpolate values for test input grid parameters.

which contain the beta function parameters, namely AUC, α, and β. The output grid of parameter estimates are used to draw a contour map overlapping the biplot data. A contour map for the alpha parameter output variable, using 15Y maturity swap data is plotted in figure 5.2.

The contour map provides information, not only on the interpolation biplot input parameters, but also of the output parameter values for a specific parameter. The peaks in the contour map can be used to visualise the relationship between the output parameter and the input parameters. For a new test input data point plotted on the contour map, an estimate of the output parameter value can be computed. This is done in figure 5.2, where the interpolated test output parameter estimate is plotted on the contour map, as indicated by the red data point. This interpolated test output parameter corresponds to a test input with values: $L = 0.0816$, $S = -0.0319$, $C = 0.0169$, $T = 15$, and $r_{fix} = 0.11$.

In this section, biplots and contour maps were plotted for the training grid data. Plots play an important role in providing a snapshot of complex large data sets, which aids in comprehension of the relationships within the data sets. In the following section, the methodology is repeated for a large set of within-grid and historical NS parameter test data.

5.2 TESTING WITHIN-GRID DATA PERFORMANCE

In this section, the data used to test the biplot interpolation method will be discussed. Within-grid data is randomly chosen for values which lie within the input grid boundary values for each of the input parameters (L, S, C, r_{fix}, and T). An excerpt of the sample data ordered according to maturity is given in table 5.1.

	$\hat{L}$	$\hat{S}$	$\hat{C}$	T	r_{fix}
1	0.0887	-0.0255	0.0283	1.2276	0.0487
2	0.0873	0.0074	-0.0339	1.2358	0.0399
...	...	...	...	...	...
99	0.0750	0.0016	-0.0311	28.2316	0.1155
100	0.0827	0.0135	-0.0327	29.6822	0.0657

Table 5.1: An excerpt of the within-grid test input grid

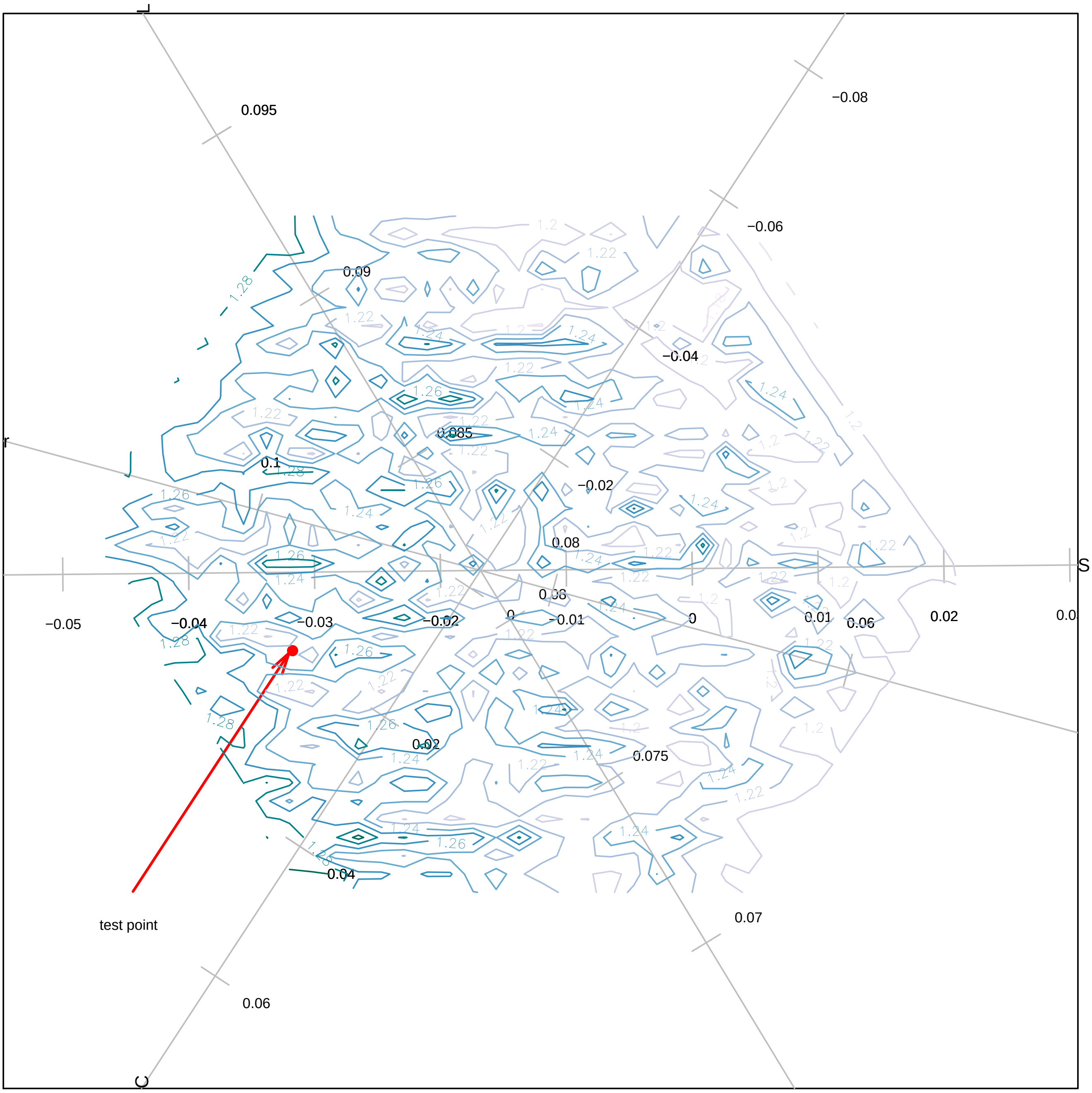


Figure 5.2: α parameter contour plot for 15Y maturity data with test sample

A contour map of the $\hat{\alpha}$ values are plotted overlaying the input biplot plot in figure 5.1. The link between the $\hat{\alpha}$ values and the input grid values can be seen by the various peaks on the plot. A test input sample with parameter values: $L = 0.0816$, $S = -0.0319$, $C = 0.0169$, $T = 15$, and $r_{fix} = 0.11$, is plotted. Using interpolation the output α parameter estimate can be computed.

In the following sections the error results and a plot of the output for various test input data values are included. The piecewise-linear interpolation method was used to estimate the beta parameter estimates for the within-grid data discussed in this section. The estimated EE values using the biplot interpolation method are compared to the true expected exposure values within standard deviation bands of 1.5, 1, and 0.5 standard deviations. The average errors are the percentage of points which do not fall within the given bands at the various standard deviation bands. The average error results are recorded in table 5.2.

5.2.1 Error results

The number of data points used for maturities between one to five years is small, and therefore the average errors are more variable as compared to the other maturities which contain more data points. In particular, at the 1.5SD and 1SD bands the average errors are much higher as compared to the other maturity ranges. The average errors for the one to five year swaps is approximately 20% for the 1.5SD and 1SD bands, whereas the average errors for the other maturity ranges are all below 12%.

As the maturity range increases, the average error for each standard deviation band has an increasing trend, where average errors for lower maturities have smaller average errors than the higher maturities. This result could be due to higher maturities having a larger number of data points than the lower maturities and therefore an increased probability of error in estimation. However, this result does not apply to the maturity range of one to five years.

As the standard deviation band decreases, the number of biplot interpolation estimated expected exposure points which do not fall within the band increases, i.e. the error increases. Overall the average errors for data points which fall outside the 0.5SD, 1SD, and 1.5SD bands are: 25.3167%, 8.8017%, and 5.9317%.

The results shown in table 5.2 are only representative of a single random sample of 100 points. Therefore, due to sampling variability, the errors can vary between different draws. An average of various random draws can be computed to provide a more stable representation of the average errors for each maturity range.

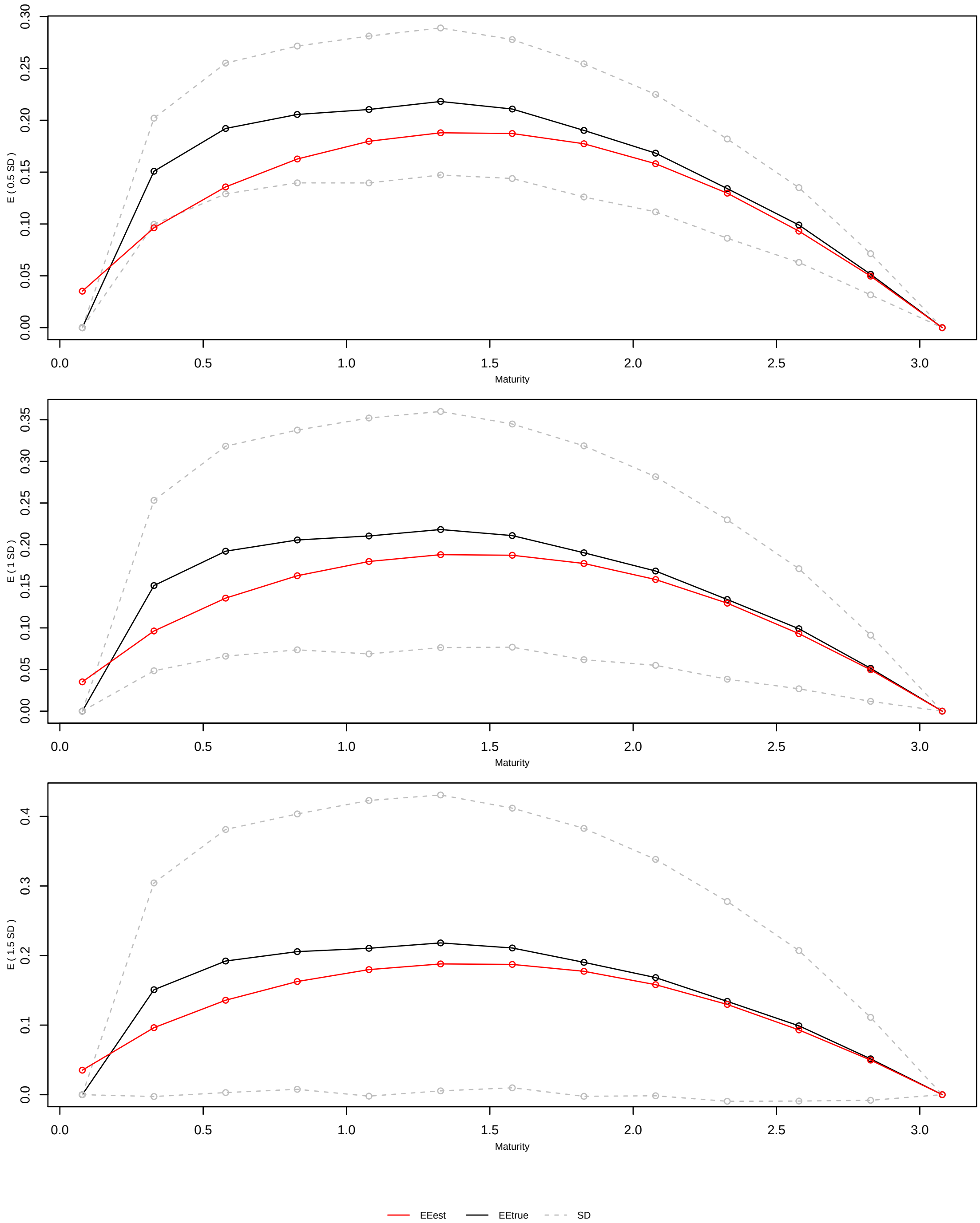


Figure 5.3: Estimated Exposure Estimations Output

The expected exposure is estimated for a sample with input parameters: $L = 0.0811$, $S = -0.0227$, $C = 0.0056$, $T = 3.0783$, and $r_{fix} = 0.0808$. The biplot estimation of the EE is represented by the red curve, the true EE is represented by the black curve and the grey curves represent the SD of the exposure estimations. Within a standard deviation band of 0.5SD, two points (15.38%) fall outside the bands. For standard deviation bands of 1SD and 1.5SD, only the first expected exposure estimate (7.69%) falls outside the standard deviation band.

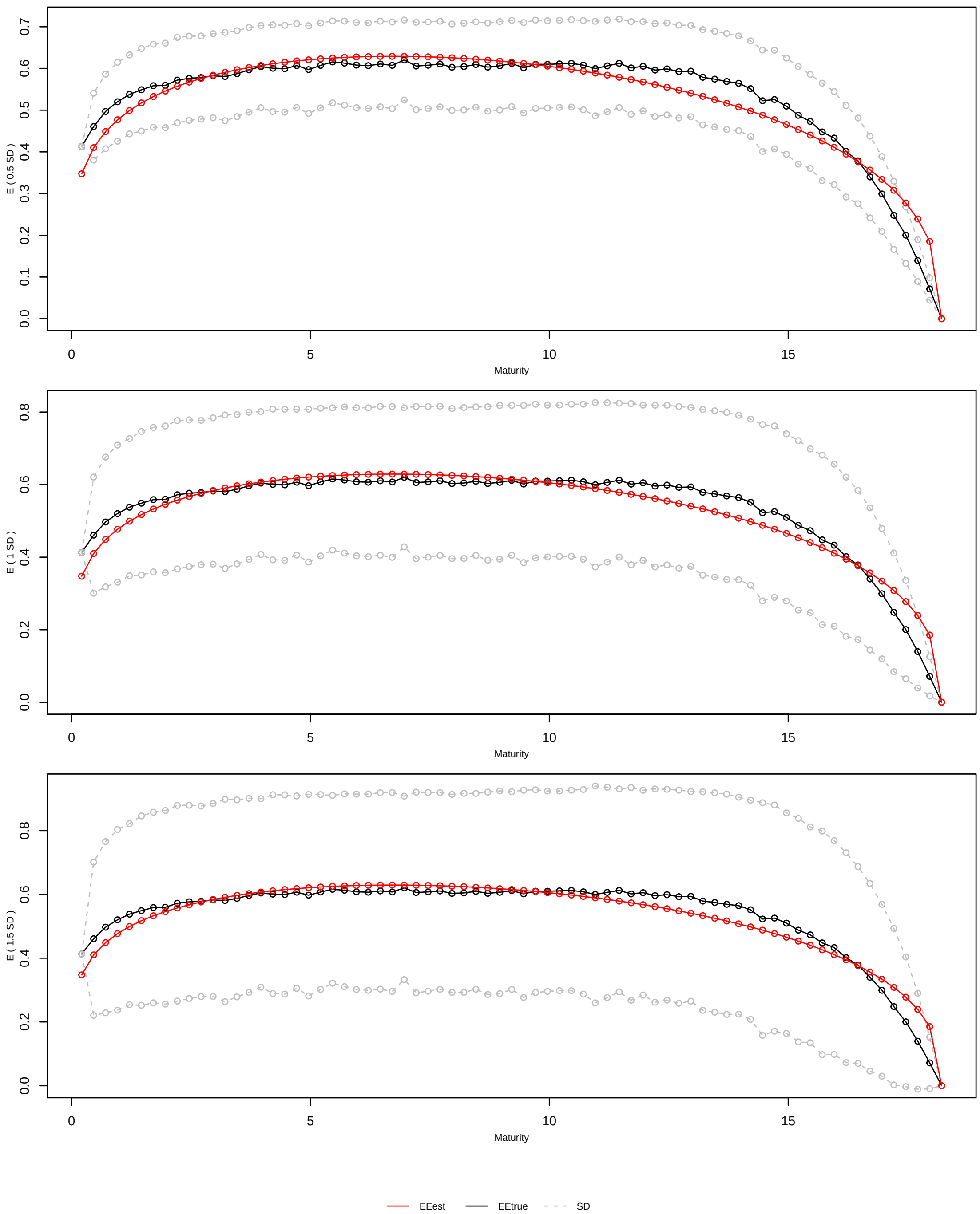


Figure 5.4: Estimated Exposure Estimations Output

The expected exposure is estimated for a sample with input parameters: $L = 0.0816$, $S = -0.0319$, $C = 0.0169$, $T = 18.2111$, and $r_{fix} = 0.1145$. The biplot estimation of the EE is represented by the red curve, the true EE is represented by the black curve and the grey curves represent the SD of the exposure estimations. For a 0.5SD band, 5.48% of the estimated EE points fall outside the SD bands, and for 1SD and 1.5SD bands, 2.74% of the estimated EE points fall outside the SD bands.

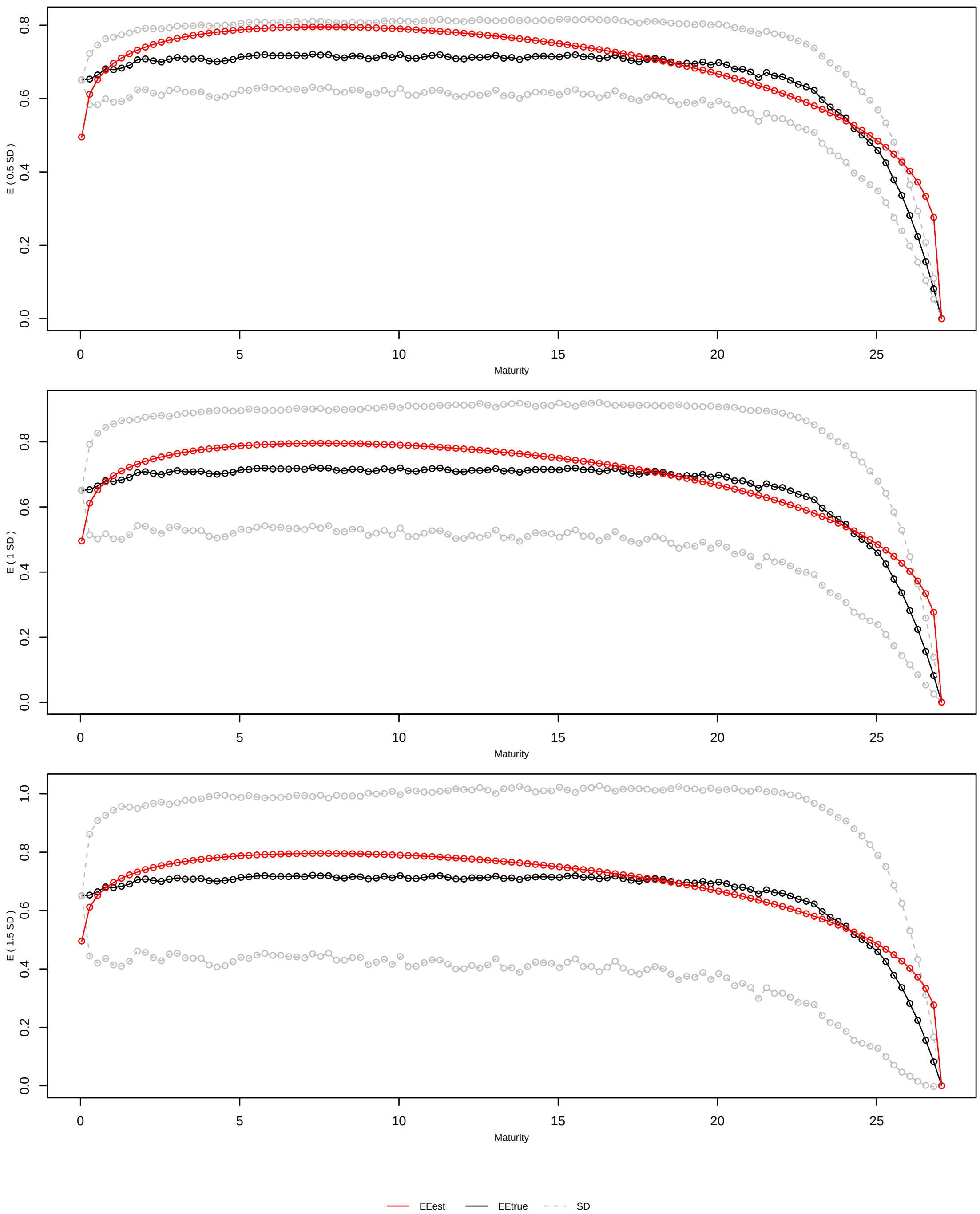


Figure 5.5: Estimated Exposure Estimations Output

The expected exposure is estimated for a sample with input parameters: $L = 0.0827$, $S = -0.0197$, $C = 0.0046$, $T = 27.0392$, and $r_{fix} = 0.0761$. The biplot estimation of the EE is represented by the red curve, the true EE is represented by the black curve and the grey curves represent the SD of the exposure estimations. For a 0.5SD band, 4.59% of the estimated EE points fall outside the SD bands. For a 1SD band, 3.67% of the estimated EE points fall outside the SD bands, and for 1.5SD bands, 2.75% of the estimated EE points fall outside the SD bands.

Maturity	No. Data Points	Average Errors		
		1.5 SD	1 SD	0.5 SD
1-5	15	0.2000	0.2039	0.2492
5-10	16	0.0375	0.0432	0.1274
10-15	16	0.0256	0.0341	0.1815
15-20	13	0.0282	0.0673	0.2743
20-25	22	0.0256	0.0602	0.2977
25-30	18	0.0390	0.1194	0.3889

Table 5.2: Average EE error results for within-grid test data

5.2.2 Output graph examples of EE within confidence intervals

In figures 5.3 to 5.5, plots of the estimated EE profiles are plotted at 1.5, 1, and 0.5 standard deviations for various samples.

In the following section, historical NS parameter data is used to test the biplot interpolation method.

5.3 TESTING HISTORICAL NS PARAMETER DATA PERFORMANCE

Historical NS parameter data is chosen at random for values which include historical NS parameter estimates (L, S, and C), and within grid boundary values for the rate and maturity parameters (r_{fix} and T). An excerpt of the historical NS parameter data is given in table 5.3.

	$\hat{L}$	$\hat{S}$	$\hat{C}$	T	r_{fix}
1	0.0730	0.0443	-0.0242	1.1023	0.1106
2	0.0779	0.0265	-0.0249	2.1826	0.1283
...	...	...	...	...	...
99	0.0828	-0.0177	0.0000	29.7380	0.0789
100	0.0963	-0.0471	-0.0388	29.7415	0.0492

Table 5.3: An excerpt of the historical NS parameter test input grid

In the following sections the error results and a plot of the output for various test input data values are included. In discussing the error results, further detail is provided for input test data which falls outside the plotted input biplot regions, this is covered in section 5.3.2.

5.3.1 Error results

The interpolation of the output parameter estimates using piecewise-linear interpolation is highly sensitive to the input grid values (training data) used, as values which fall outside the convex hull of input grid values cannot be estimated. Due to this shortcoming of the piecewise-linear interpolation method, two interpolation methods will be implemented for historical NS parameter data, namely piecewise-linear interpolation without extrapolation of values, and Akima splines interpolation with extrapolation of values.

5.3.1.1 Piece-linear interpolation

Firstly, piecewise-linear interpolation of the output parameter estimates will be implemented. The piecewise-linear interpolation method used is barycentric interpolation, which uses Delaunay triangulation (Delaunay *et al.*, 1934) of the input values. Firstly, the input points are plotted, and the corresponding convex-hull is divided into Delaunay triangles. The triangles are created using the $n_{dim} + 1$ nearest neighbours in the plotted space. As an example, a Delaunay triangular division of input points is plotted in figure 5.6 for two dimensional data. Similarly as in figure 5.6 the Delaunay triangular division of input points, can be applied to higher dimensions.

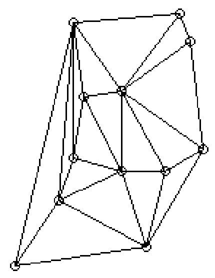


Figure 5.6: Input grid divided into Delaunay triangles

For a new test observation, the approximate value of the test observation, is calculated using a weighted average of the closest $n_{dim} + 1$ neighbours as determined by the Delaunay triangulation of the convex hull of input points. Barycentric coordinates are then used to assign weights to the closest $n_{dim} + 1$ neighbours. The test observation (x) which falls within the convex of points (x_1, x_2, x_3, and x_4) can be written as:

$$x = \sum_{i=1}^{n_{dim}+1} \alpha_i x_i \tag{5.1}$$

where, α_i are the barycentric coordinates and the constraint $\sum_{i=1}^{n_{dim}+1} \alpha_i = 1$ must hold.

The interpolation function, $g(.)$, used to approximate the value of a new test observation is defined as:

$$g(x) \approx \sum_{i=1}^{n_{dim}+1} \alpha_i g(x_i) \tag{5.2}$$

For a test point which does not fall within the convex hull of input points, the barycentric interpolation method cannot be used to approximate the test point value, as the point will not fall within a Delauanay triangle which is used to approximate the values.

In the piecewise-linear interpolation case, output parameter estimates cannot be calculated for test input values which do not fall within the set (convex hull) of input grid values (training data). Therefore no extrapolation of values for test input values which fall outside the convex hull can be calculated using piecewise-linear interpolation, and an NA value is returned for these test input values. In table 5.4, the error values for input test data calculated using piecewise-linear interpolation are recorded. In table 5.4 the number of for which an estimate value cannot be extrapolated (i.e. NA values) using piecewise-linear interpolation are indicated in brackets.

The output parameter estimates for the test input data values which cannot be estimated using piecewise-linear interpolation are assigned boundary values for each output parameter. The boundary values are set equal to zero for each of the output parameter estimates ($A\hat{U}C$, $\hat{\alpha}$, and $\hat{\beta}$). The resulting expected exposure profile for these cases is floored at zero. The resulting error results adjusted to include the boundary conditions are recorded in table 5.5.

In the following subsection, Akima splines interpolation is used to estimate the output parameters.

Maturity	No. Data Points	Average Errors		
		1.5 SD	1 SD	0.5 SD
1-5	12 (2)	0.1358	0.158	0.2435
5-10	15 (2)	0.0359	0.041	0.1239
10-15	16 (3)	0.0259	0.0401	0.2341
15-20	19 (4)	0.0299	0.0518	0.2779
20-25	22 (4)	0.0266	0.0609	0.3138
25-30	16 (3)	0.0259	0.0683	0.2606

Table 5.4: Average EE error results for historical NS test data using piecewise-linear interpolation

Maturity	No. Data Points	Average Errors		
		1.5 SD	1 SD	0.5 SD
1-5	12	0.2798	0.2983	0.3695
5-10	15	0.1644	0.1688	0.2407
10-15	16	0.2085	0.2201	0.3777
15-20	19	0.2342	0.2514	0.4300
20-25	22	0.2036	0.2317	0.4385
25-30	16	0.2086	0.2430	0.3993

Table 5.5: Average EE error results for historical NS test data using piecewise-linear interpolation and boundary conditions

5.3.1.2 Akima splines interpolation

In the Akima splines interpolation method, extrapolation of data values can be estimated. For a test input, the approximation of the point values using Akima splines is defined by the interpolation function, h, as (Akima, 1970):

$$h(x) = a_0 + a_1.(x - x_i) + a_2.(x - x_i)^2 + a_3.(x - x_i)^3,\ x_i \leq x \leq x_{i+1} \tag{5.3}$$

where, $h_i = h(x_i)$, $1 \leq i \leq k$, are the input points, and a_0, a_1, a_2, and a_3 are coefficients of the interpolation polynomial for each interval $[x_i, x_{i+1}]$.

The coefficient values are determined using the function values h_i and h_{i+1}, and the first derivatives h'_i and h'_{i+1} at the end points of the intervals.

The Akima splines interpolation method is sensitive to the input grid values used, as the calculated output parameter estimates for test input data which falls far outside the input grid values do not provide reasonable results. The error results calculated using Akima splines interpolation are recorded in table 5.6. In the historical NS parameter data, four data points calculated using Akima splines interpolation result in unreasonable (NaNs) estimates for the beta function parameters. These data values are discussed in detail in section 5.3.2. In the table 5.6 below, the number of test data points which result in NaNs values for the output parameter estimates are indicated in brackets.

Maturity	No. Data Points	Average Errors		
		1.5 SD	1 SD	0.5 SD
1-5	12(2)	0.1558	0.1558	0.2258
5-10	15(1)	0.1606	0.2132	0.3879
10-15	16(1)	0.0628	0.1680	0.4580
15-20	19	0.1514	0.2809	0.5428
20-25	22	0.1434	0.2383	0.5360
25-30	16	0.2026	0.3037	0.5406

Table 5.6: Average EE error results for historical NS test data using Akima splines interpolation

The output parameter estimates for the s which result in unreasonable estimates using Akima splines interpolation are assigned boundary values for each output parameter. The boundary values are set equal to zero for each of the output parameter estimates ($\widehat{AUC}$, $\hat{\alpha}$, and $\hat{\beta}$). The resulting error results adjusted to include the boundary conditions are recorded in table 5.7.

The average errors using extrapolated data are significantly higher than the average errors calculated using linear interpolation. Although a higher number of estimates for the output parameters can be obtained using the extrapolation method as compared to the linear interpolation method, the estimated results using extrapolation are not as accurate as compared to the linear interpolation method.

Maturity	No. Data Points	Average Errors		
		1.5 SD	1 SD	0.5 SD
1-5	12	0.2965	0.2965	0.3548
5-10	15	0.2166	0.2656	0.4287
10-15	16	0.1214	0.2200	0.4919
15-20	19	0.1514	0.2809	0.5428
20-25	22	0.1434	0.2383	0.5360
25-30	16	0.2026	0.3037	0.5406

Table 5.7: Average EE error results for historical NS test data using Akima splines interpolation and boundary conditions

5.3.2 Extrapolation error results

The extrapolation errors are investigated for data values which fall outside the input biplot regions. For one to five year maturities the errors corresponding to each of the 12 test data are compared in table 5.8 for a 0.5SD band. The second column refers to error results obtained using the linear interpolation method, and the third column refers to error results obtained using Akima splines interpolation with extrapolation.

Data Point	Piecewise-linear interpolation	Akima splines interpolation
1	NA	NA
2	0.6667	0.1111
3	0.8	0.3
4	NA	NA
5	0.1	0.9
6	0.0833	0.0833
7	0.1429	0.0714
8	0.0667	0.0667
9	0.0625	0.0625
10	0.0625	0.0625
11	0.15	0.35
12	0.3	0.25

Table 5.8: Maturities 1-5 years average error results with 0.5SD band

In the table 5.8 above, the error values for test data point 1 and test data point 4 result in NaNs values when extrapolating the input data values. These test data points are studied further. The test data points correspond to the input values in table 5.9.

	$\hat{L}$	$\hat{S}$	$\hat{C}$	T	r_{fix}
1	0.0730	0.0443	-0.0242	1.1023	0.1106
4	0.0723	0.0145	0.0372	2.2857	0.0824

Table 5.9: Historical NS parameter input data

The input biplot used in extrapolating the values for the input test data is plotted in figure 5.7 below. It can be seen that the input test data values from test data point 1 and test data point 4, which correspond to 1.1023Y and 2.2857Y swaps fall outside the biplot plotting region.

The beta function interpolated values corresponding to test data points, 1 and 4, are recorded in table 5.10 below:

Data Point	AUC	α	β
1	-155.7652	-3.4063	-0.4987
4	-969,6672	-11,6454	-6,4997

Table 5.10: Beta function extrapolated parameter estimates

The estimated beta function parameters result in a NaNs value when computing the constant value for the Euler integral, with $Const = \widehat{AUC} \times b(\hat{\alpha}, \hat{\beta})$, where the Euler integral cannot be calculated for negative values of $\hat{\alpha}$, and $\hat{\beta}$.

In the cases where the extrapolated values result in NaNs values, the estimated values for the parameters are set equal to the boundary values of the AUC, α and β parameters, and the resulting EE profile lies flat at zero. In figure 5.8, the adjusted EE profile is plotted for test data point 1 using the boundary condition.

5.3.3 Output graph examples of expected exposure within confidence intervals

In figures 5.9 to 5.11, plots of the estimated EE profile using the linear interpolation method are plotted for various historical NS parameter data samples, at 0.5, 1, and 1.5 standard deviations.

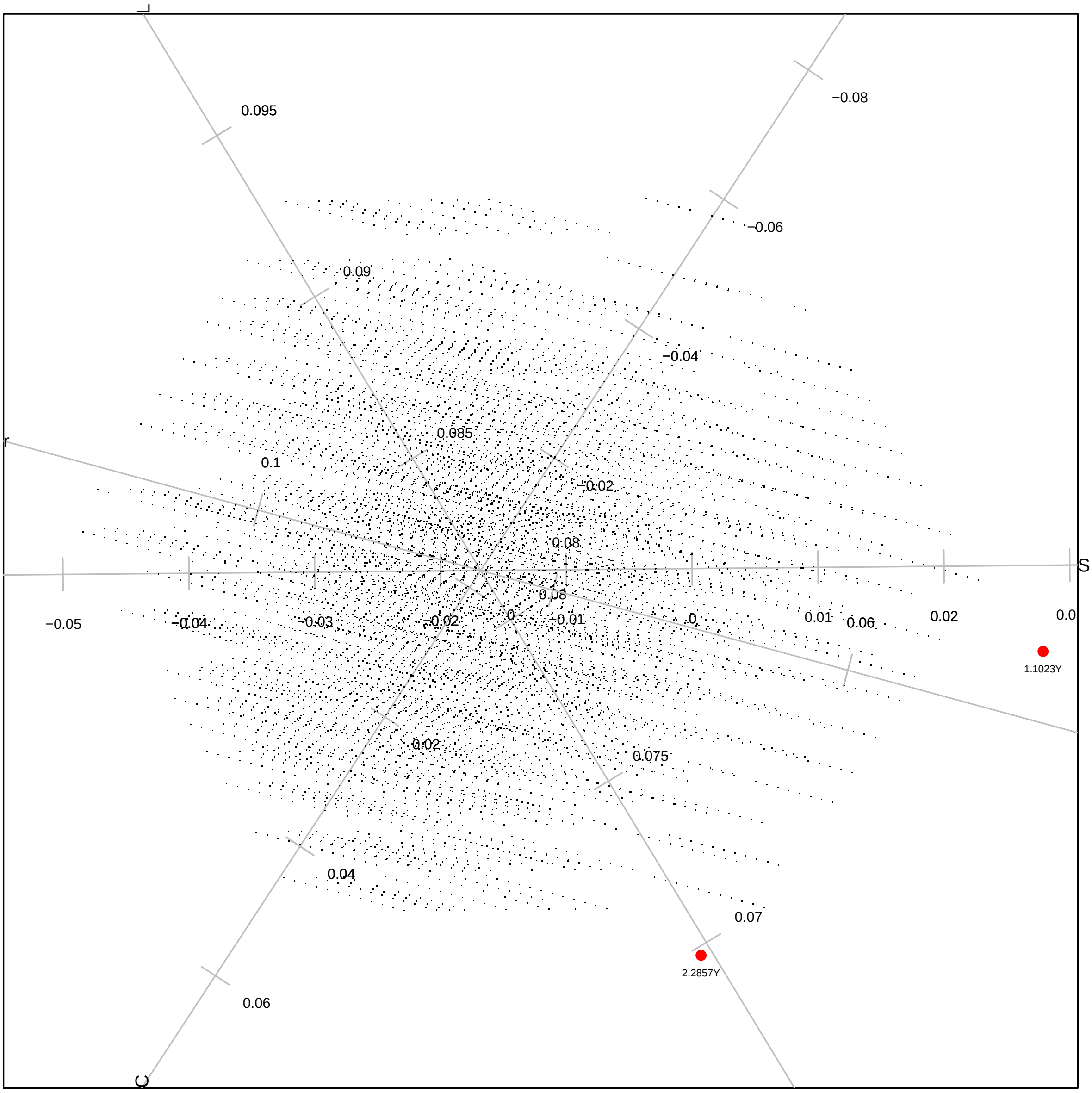


Figure 5.7: Input biplot with historical NS parameter test data

The 1Y input biplot is plotted, along with the historical NS parameter input test parameters which correspond to test data point 1 and test data point 4. Test data point 1 is a 1.1023Y swap with input parameters: $L = 0.0730$, $S = 0.0443$, $C = -0.0242$, and $r_{fix} = 0.1106$ and test data point 4 is a 2.2857Y swap with input parameters: $L = 0.0723$, $S = 0.0145$, $C = 0.0372$, and $r_{fix} = 0.0824$. It can be seen on the biplot that the 1.1023Y, and 2.2857Y swaps fall outside the range of the input biplot observations. These two test swaps result in extrapolated values for the output beta function parameters which are not within the bounded values of the beta function distribution. Optionally, a contour map can also be plotted, which would also show that the estimates for the output test test data point 1 and test data point 4 do not fall within the contour map region of estimated output values.

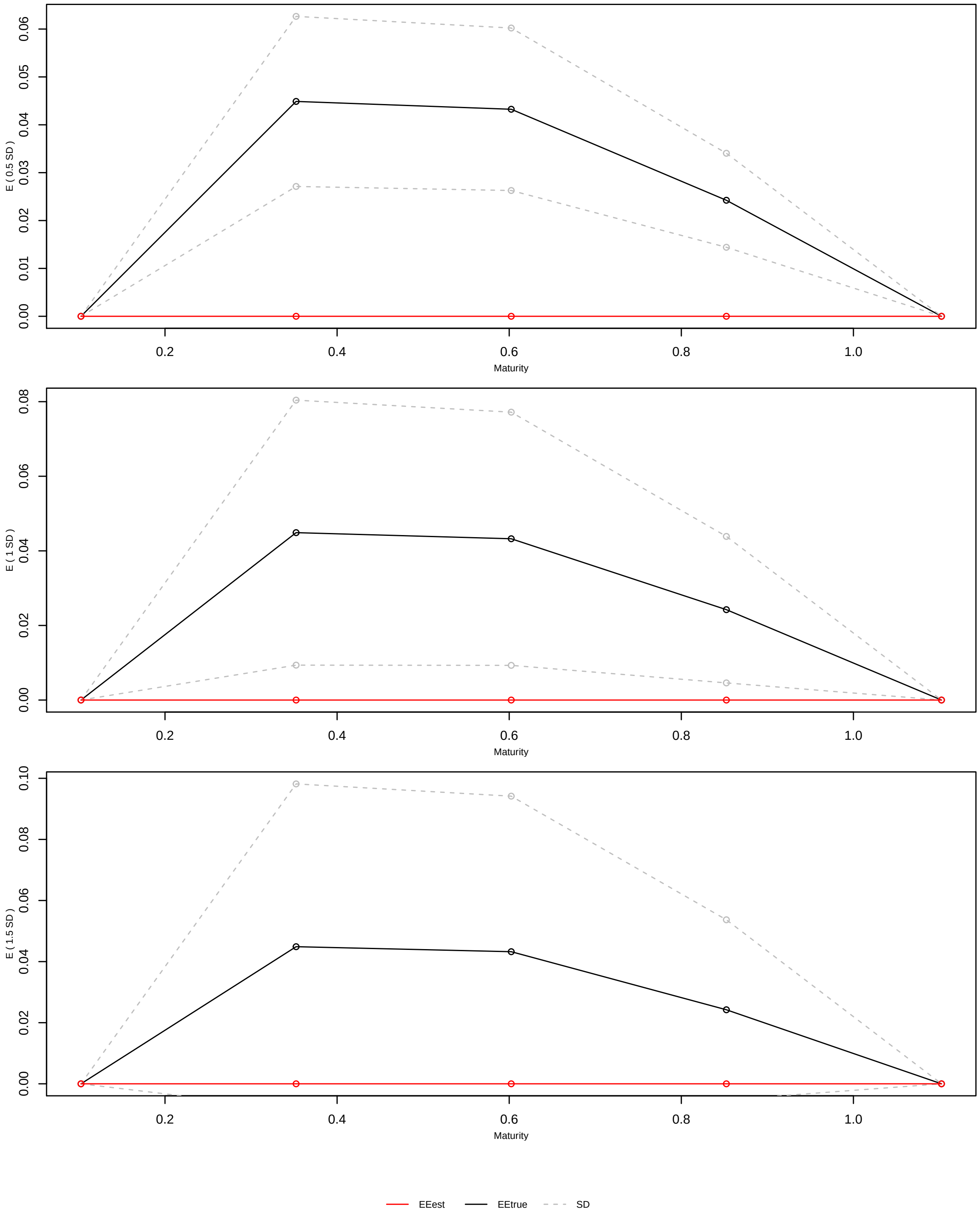


Figure 5.8: Input biplot with historical NS parameter test data

The estimated expected exposure distribution is plotted for test data point 1, using the boundary conditions for the output parameter estimates. Test data point 1 is a 1.1023Y swap with input parameters: $L = 0.0730$, $S = 0.0443$, $C = -0.0242$, and $r_{fix} = 0.1106$. It can be seen that for all values of the EE the profile lies flat at zero.

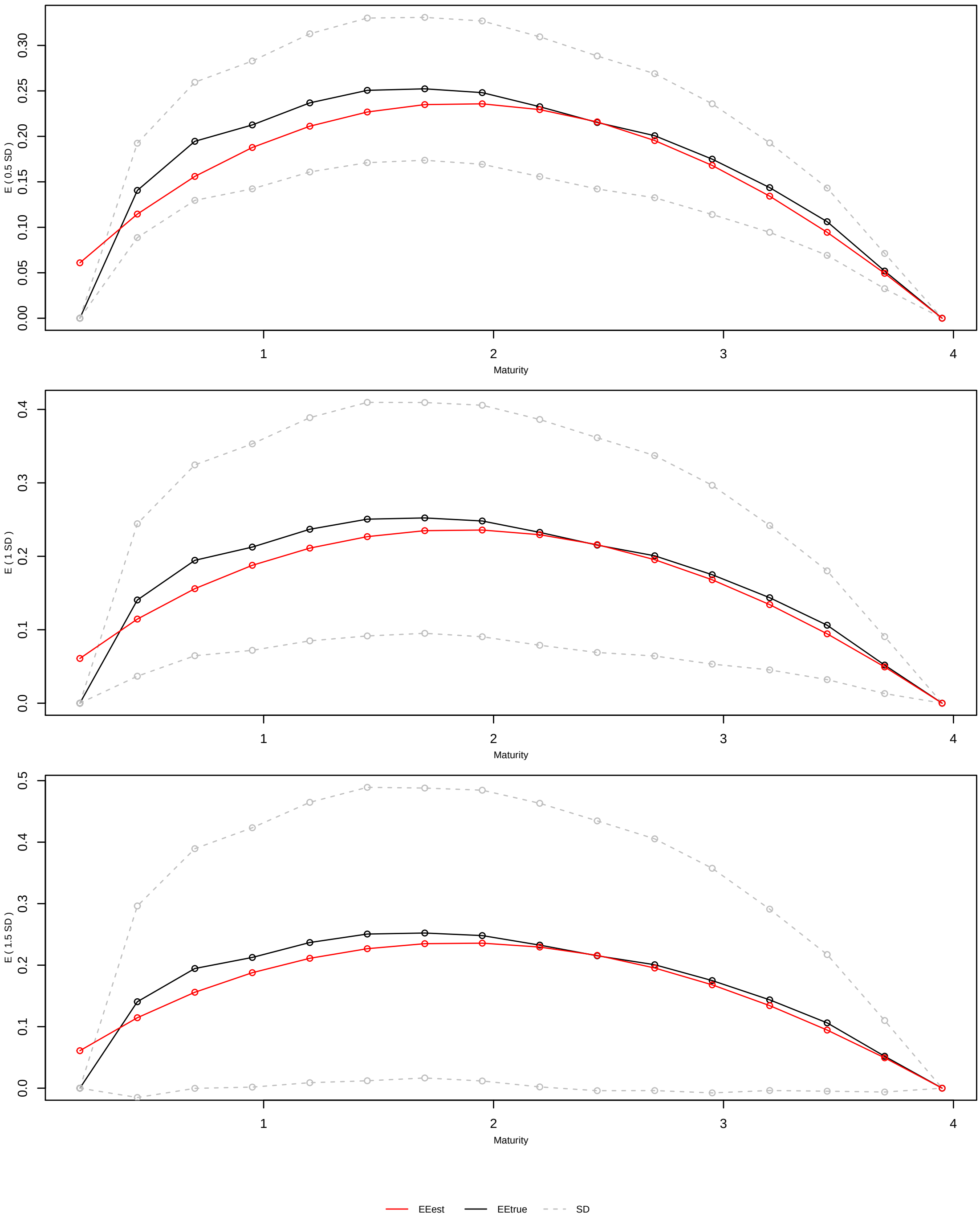


Figure 5.9: Estimated Exposure Estimations Output

The expected exposure is estimated for a sample with input parameters: $L = 0.0917,\ S = -0.0303, C = -0.0084,\ T = 3.9508,$ and $r_{fix} = 0.1212$. The biplot estimation of the EE is represented by the red curve, the true EE is represented by the black curve and the grey curves represent the SD of the exposure estimations. At each SD band, the first estimated EE point (6.25%) falls outside the SD bands.

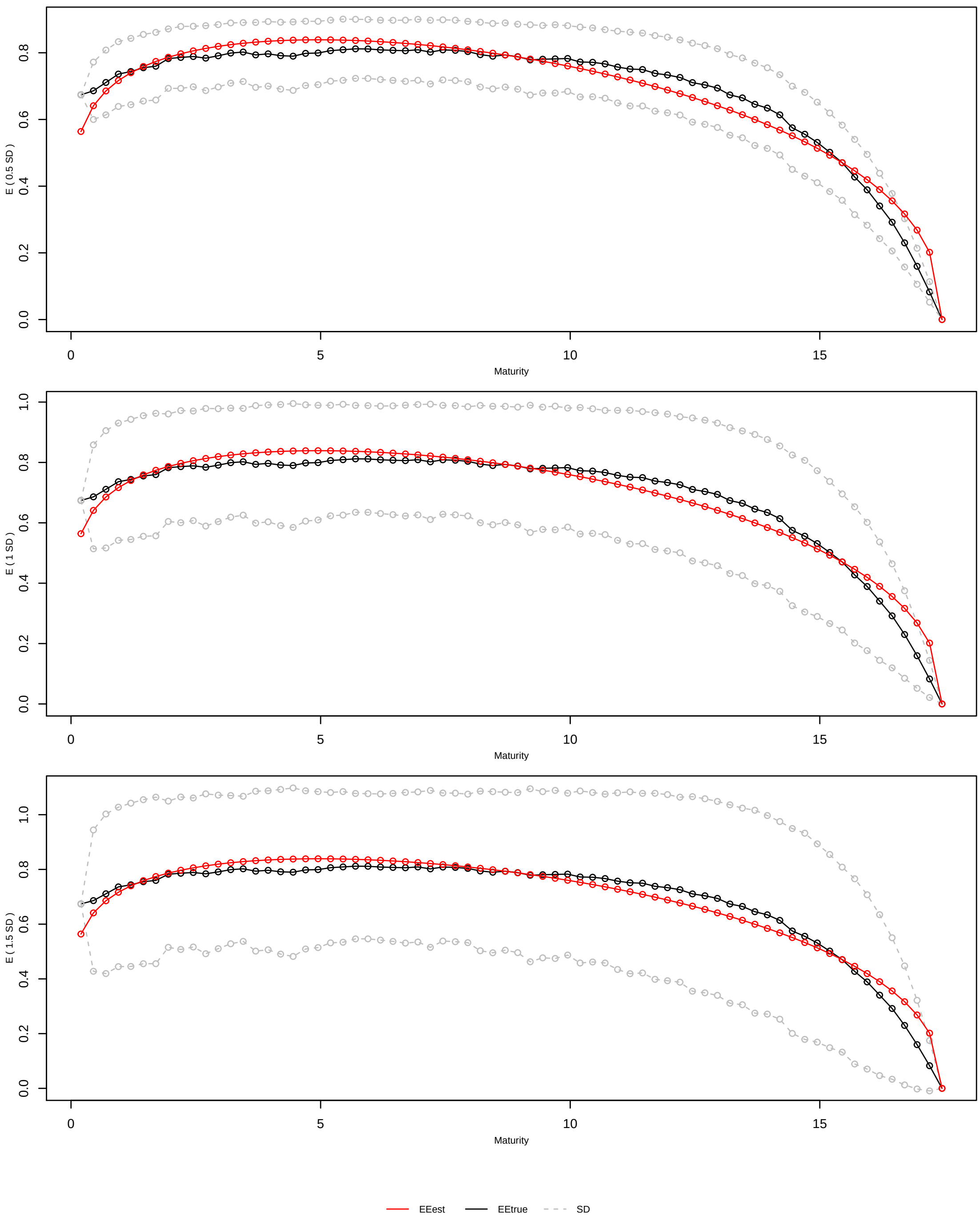


Figure 5.10: Estimated Exposure Estimations Output

The expected exposure is estimated for a sample with input parameters:$L = 0.0742$, $S = 0.0176$, $C = 0.0022$, $T = 17.4488$, and $r_{fix} = 0.0362$. The biplot estimation of the EE is represented by the red curve, the true EE is represented by the black curve and the grey curves represent the SD of the exposure estimations. For a 0.5SD band, 5.71% of the estimated EE points fall outside the SD bands. For a 1SD band, 4.29% of the estimated EE points fall outside the SD bands, and for 1.5SD bands, 2.86% of the estimated EE points fall outside the SD bands.

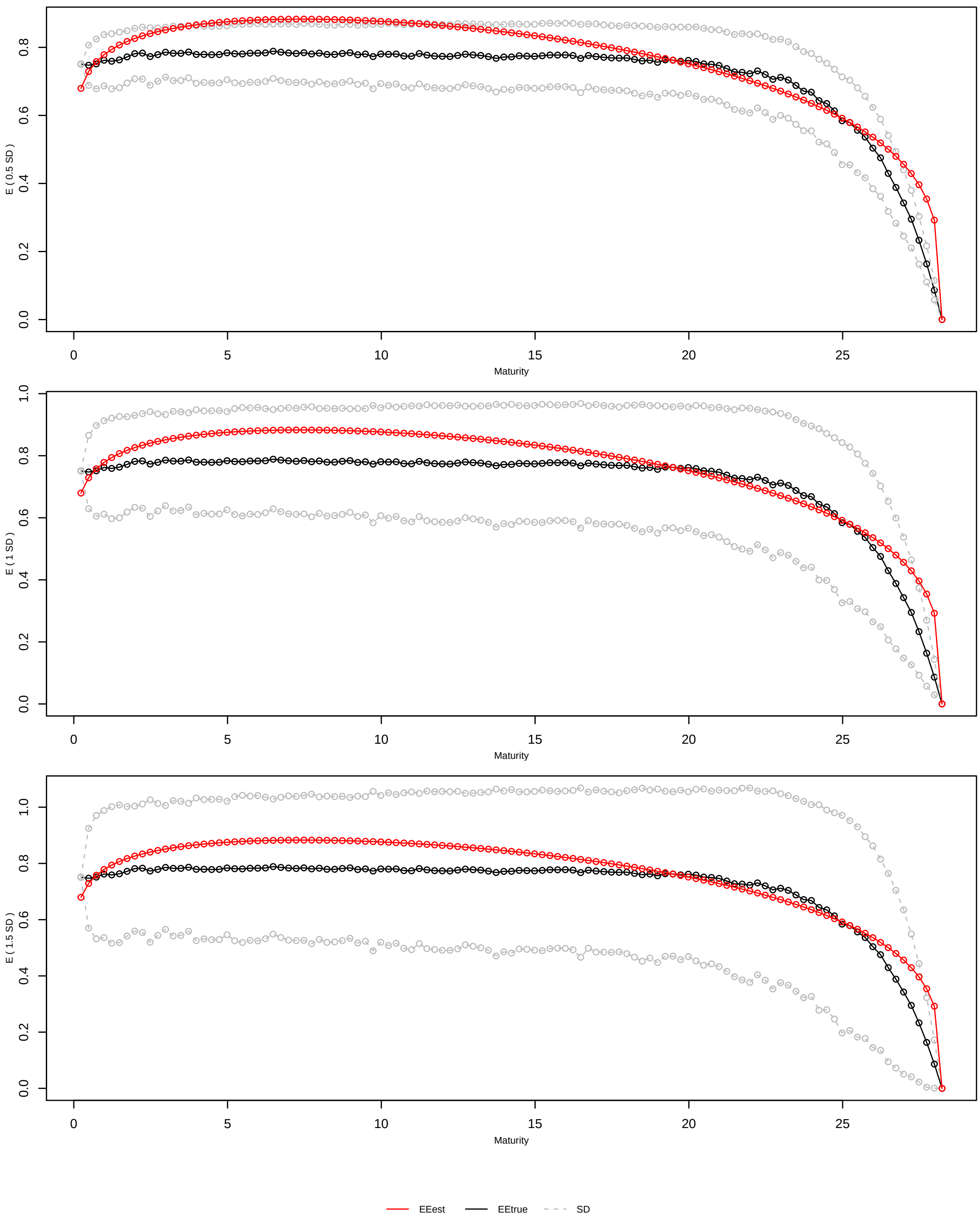


Figure 5.11: Estimated Exposure Estimations Output

The expected exposure is estimated for a sample with input parameters: $L = 0.0749$, $S = -0.0048$, $C = 0.0733$, $T = 28.233$, and $r_{fix} = 0.0519$. The biplot estimation of the EE is represented by the red curve, the true EE is represented by the black curve and the grey curves represent the SD of the exposure estimations. For a 0.5SD band, 31.86% of the estimated EE points fall outside the SD bands. For a 1SD band, 3.54% of the estimated EE points fall outside the SD bands, and for 1.5SD bands, 2.65% of the estimated EE points fall outside the SD bands.

5.4 SUMMARY

In this chapter, the biplot interpolation method was tested using biplot approximated EE data for both within-grid and historical NS parameter data. The within-grid data estimates performed relatively well, where calculated errors for each maturity range was within 8,8017% for a 1SD interval.

The historical NS parameter data estimates did not perform as well as the within-grid data. For historical NS parameter data using linear interpolation, the error results were in line with those of the within-grid data, where calculated errors for each maturity range was within 7,0017% for a 1SD interval. The linear interpolation approach cannot extrapolate values for points which fall outside the convex hull. As a result, error results could only be recorded for 88 out of the 100 test data points using the linear interpolation method. The within-grid sample and historical NS parameter performance cannot be compared as the number of used in the average error calculations were not equal.

Extrapolation of values was considered using the Akima splines interpolation method. The average errors recorded using the Akima splines interpolation method were much higher as compared to both the linear interpolation method using historical NS parameter data, and the linear interpolation method using within-grid data. The biplot interpolation method is sensitive to input values which fall outside of the input grid range of values, and extrapolation of values which fall outside the input grid result in large errors, i.e. the resulting EE estimates are highly inaccurate.

The extrapolation errors could be decreased by increasing the size of the input grid used. A larger input grid would contain a larger range of values and the input test data would fall closer within the range of input values resulting in a more accurate interpolated/extrapolated estimate. Further conclusions and recommendations for further studies are made in the final chapter.

CHAPTER 6

CONCLUSION

The main aim of this study was to implement a biplot interpolation approach to estimate beta function parameters to be used to approximate EE. This was done by firstly studying the necessary background theory on EE and PCA biplots. Firstly, the various methods used to estimate EE, and the importance of estimating exposure for risk management purposes, were discussed. Thereafter, PCA biplots and their interpretation were studied, where the theoretical background used to construct biplots was also detailed.

Thereafter, a biplot interpolation method was proposed in the methodology chapter. The biplot interpolation method is used to link input and output data parameters. The link between the input and output parameters is studied using biplots and tri-linear interpolation. An application of the biplot interpolation methodology to IRS data was discussed in detail. The application of the biplot interpolation methodology started with the creation of an input grid containing IRS parameters. Thereafter, the simulation of the output grid which contained beta function parameters was discussed. In order to estimate the beta function parameters, the EE values for each input grid IRS was simulated using the MC simulation approach.

With the application of the biplot interpolation method discussed in detail, the methodology was then tested using a standard deviation band approach. The testing of the biplot interpolation method using IRS data formed one of the main components of this study. Two sets of data were used to test the methodology, namely data which was simulated for values which were within the input grid boundary values (within-grid data), and data which was simulated using historical data, (historical NS parameter data). It was noted that, the input grid used to set up the interpolation method played a crucial role in the accuracy of the estimated output values. For test data which fell far from the initial input grid values, the interpolation methods used, did not provide reasonable estimates.

It is recommended that different interpolation methods to estimate output parameters for three-variable data also be considered. Furthermore, the inputs used in the model to estimate EE were not all modeled using historical data, where some of the inputs were chosen arbitrarily.

A standard deviation band test approach was used to test the validity of the biplot interpolation approach. For within-grid test data, the estimated exposure curves provided a reasonable estimate for the true EE profiles, where the MC simulation approach was used as a proxy to estimate the true EE profiles. The historical NS parameter test data did not perform as well as the within-grid test data, where the extrapolation of output data was not straightforward to implement, and the results were not always reasonable.

The biplot interpolation method was studied as an alternative to the commonly used MC simulation approach, used to estimate EE . The MC approach is computationally expensive due to the number of simulations required to estimate the EE profile. The complexity of the MC approach is dependent on the model parameters used to simulate the underlying interest rate curve, as well as the input variables used to accurately estimate EE . The aim of the biplot interpolation approach was to provide a simpler approach to estimate the EE profile. As compared to the MC simulation approach, the biplot interpolation approach only requires the estimation of beta function parameters in order to estimate the EE profile. A drawback of the biplot interpolation approach is that it requires upfront work to set up the input and output grids. However, once the input and output grids have been set up, the biplot interpolation approach is quick to apply, and does not require large numbers of simulations as in the MC case.

In conclusion, the biplot interpolation approach proposed in this study provided an interesting application of biplots as a tool to study financial data.

REFERENCES

Abramowitz, M. and Stegun, I.A. (1948). *Handbook of mathematical functions with formulas, graphs, and mathematical tables*, vol. 55. US Government printing office.

Akima, H. (1970). A new method of interpolation and smooth curve fitting based on local procedures. *Journal of the ACM (JACM)*, vol. 17, no. 4, pp. 589–602.

Antwi, O. (2014). Measuring portfolio loss using approximation methods. *Science Journal of Applied Mathematics and Statistics*.

Consiglio, A. and Guirreri, S. (2011 September). Simulating the term structure of interest rates with arbitrary marginals. *International Journal of Risk Assessment and Management*, vol. 15, no. 4.
Available at: `http://dx.doi.org/10.1504/IJRAM.2011.042670`

Delaunay, B. *et al.* (1934). Sur la sphere vide. *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk*, vol. 7, no. 793-800, pp. 1–2.

Diebold, F.X. and Li, C. (2006). Forecasting the term structure of government bond yields. *Journal of econometrics*, vol. 130, no. 2, pp. 337–364.

Eckart, C. and Young, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, vol. 1, no. 3, pp. 211–218.

Gabriel, K.R. (1971). The biplot graphic display of matrices with application to principal component analysis. *Biometrika*, vol. 58, no. 3, pp. 453–467.

Gower, J.C. and Hand, D.J. (1995). *Biplots*, vol. 54. CRC Press.

Gower, J.C., Lubbe, S. and Le Roux, N.J. (2011). *Understanding biplots.* John Wiley & Sons.

Greenacre, M.J. (2010). *Biplots in practice.* Fundacion BBVA.

Gregory, J. (2015). *The xVA Challenge: counterparty credit risk, funding, collateral and capital.* John Wiley & Sons.

Grivas, T. (2016). *xVA: Calculates Credit Risk Valuation Adjustments.* R package version 0.8.1.
Available at: `https://CRAN.R-project.org/package=xVA`

Hull, J. and White, A. (1994). Numerical procedures for implementing term structure models i: Single-factor models. *Journal of derivatives*, vol. 2, no. 1, pp. 7–16.

Le Roux, N., Gardner, S. and Olivier, P. (2003). Biplots for displaying multidimensional financial performance data graphically. *South African Journal of Accounting Research*, vol. 17, no. 1, pp. 41–64.

le Roux, N. and Lubbe, S. (2013). *UBbipl: UNDERSTANDING BIPLOTS: DATA SETS AND FUNCTIONS*. R package version 3.0.4.
Available at: `http://www.wiley.com/go/biplots`

Nelson, C.R. and Siegel, A.F. (1987). Parsimonious modeling of yield curves. *Journal of business*, pp. 473–489.

Zhu, S.H. and Pykhtin, M. (2007). A guide to modeling counterparty credit risk. *GARP Risk Review, July/August*.

APPENDIX A

R CODE

A.1 NELSON-SIEGEL PARAMETER ESTIMATION

```
install.packages("YieldCurve")
library("YieldCurve")
NS.parameters.grid<-function(dataList,lambdaIn)
{
  zar_params_list<-all.parameters(dataList,lambdaIn)
  parameter_matrix<-do.call(rbind,zar_params_list[[1]])
  return(parameter.estimate(parameter_matrix))
}
boundary.values<-function(parameters)
{
  Quartile<-apply(parameters, 2,quantile)[c(2,4),]
  IQR<-Quartile[2,]-Quartile[1,]
  Lower<-Quartile[1,]-1.5*IQR
  Upper<-Quartile[2,]+1.5*IQR
  return(round(rbind(Lower,Upper),4))
}

plot.boxplot<-function(parameters)
{
  boxplot(parameters[,-4])
}

n.outliers<-function(parameters)
{
  x<-apply(parameters,2,boxplot,plot=FALSE)
  return( c(length(x$beta_0$out),length(x$beta_1$out),
            length( x$beta_2$out),length (x$lambda$out)))
}

parameter.estimate<-function(parameters)
{
  N<-dim(parameters)[2]
  nOutliers<-n.outliers(parameters)
  #probs<-seq(from=0.1,to =0.9,by=0.1)
  probs<-seq(from=0, to =1, by=0.1)
  result<-matrix(data=NA, nrow=length(probs),ncol=N)
  for(i in 1:N)
  {
    parameter_i<-parameters[,i]
    if(nOutliers[i]==0)
    {
      result[,i]<-quantile(parameter_i,probs=probs)
    }
    else
    {
      l<-boundary.values(parameters)[1,i]
      u<-boundary.values(parameters)[2,i]
      parameter_i<-parameters[-which(parameter_i<l |
                                      parameter_i>u),i]
      result[,i]<-quantile(parameter_i,probs=probs)
```

```
    }
  }
  return(result)
}

all.parameters<-function(dataList,lambdaIn)
{
  N<-length(dataList)
  x<-vector(mode="list",length = N)
  for(i in 1:N)
  {
    n<-length(dataList[[i]])
    for(j in 1:n)
    {
      names<-colnames(dataList[[i]][[j]])
      maturities<-as.double(names[-1])
      zero_rates<-dataList[[i]][[j]][,-1]
      x[[i]][[j]]<-Nelson.Siegel.fixed(rate=zero_rates,
                              maturity=maturities,lambdaIn)
    }
  }
  return(x)
}

Nelson.Siegel.fixed<-function (rate, maturity,lambdaIn)
{
  rate <- try.xts(rate, error = as.matrix)
  if (ncol(rate) == 1)
  rate <- matrix(as.vector(rate), 1, nrow(rate))
  pillars.number <- length(maturity)
  lambdaValues <- lambdaIn
  FinalResults <- matrix(0, nrow(rate), 4)
  colnames(FinalResults) <- c("beta_0", "beta_1", "beta_2",
                              "lambda")
  j <- 1
  while (j <= nrow(rate)) {
    InterResults <- matrix(0, length(lambdaValues), 5)
    colnames(InterResults) <- c("beta0", "beta1", "beta2",
                              "lambda", "SSR")
 {
      lambdaTemp <- lambdaIn
      InterEstimation <- .NS.estimator(as.numeric(rate[j,]),
                              maturity, lambdaTemp)
      BetaCoef <- InterEstimation$Par
      if (BetaCoef[1] > 0 & BetaCoef[1] < 20) {
        SSR <- sum(InterEstimation$Res^2)
        InterResults[1, ] <- c(BetaCoef, lambdaTemp, SSR)
      }
      else {
        InterResults[1, ] <- c(BetaCoef, lambdaValues[1], 1e+05)
      }
    }
    BestRow <- which.min(InterResults[, 5])
    FinalResults[j, ] <- InterResults[BestRow, 1:4]
    j <- j + 1
  }
  reclass(FinalResults, rate)
}
```

```
Nelson.Siegel.Rates<-function(tau,beta0,beta1,beta2,lambda){
  beta0 + beta1*(1-exp(-lambda*tau))/(lambda*tau) +
  beta2*((1-exp(-lambda*tau))/(lambda*tau) - exp(-lambda*tau))
}
```

A.2 INPUT GRID

```
param_files<-c("param_functions\\data_functions.R",
                    "param_functions\\NS_params.R",
                                    "data\\raw_data.R")
source_param_files<-apply(as.matrix(param_files),1,source)
#data
zar_data<-lapply(X=raw_data_labelled,FUN=list.data)

#grid inputs
lambdaIn<-0.7308
nsParameters<-NS.parameters.grid(zar_data,lambdaIn)
term<-c(1,seq(from=5,to=30,by=5))
rates<-seq(from=0.03,to=0.14,by=0.01)

#grid
gridfinal <- expand.grid(nsParameters[,1],
                            nsParameters[,2],nsParameters[,3],
                                                    term,rates)
```

A.3 EE FUNCTIONS

```
IRS.exposure<-function(curve_params,Ei,pay_leg_rate )
{
  cashflow_frequency<-0.25 #fixed parameter
  BuySell<-"Buy" #fixed parameter
  num_of_sims<-1000 #fixed parameter
  mean_reversion_a<-0.1 #fixed parameter
  volatility<-0.2 #fixed parameter

  sim_data<-list(num_of_sims=num_of_sims,
                    mean_reversion_a=mean_reversion_a,
                                    volatility=volatility)
  time_points<-time_points.calc(Ei,cashflow_frequency)
  spot_curve<-Nelson.Siegel.Rates(time_points,
                                beta0 = curve_params$beta0,
                                beta1 = curve_params$beta1,
                                beta2 = curve_params$beta2,
                                            lambda = 0.7308)
  discount_factors<-dfactors(as.matrix(spot_curve), time_points)
  trades<-list(list(Ei=Ei,pay_leg_rate=pay_leg_rate,
                                            BuySell=BuySell))

  result_EE<-CalcSimulatedExposure_Adj(discount_factors =
                                        discount_factors,
                                        time_points = time_points,
                                        spot_curve = spot_curve,
                                        trades=trades,
                                        sim_data = sim_data)
```

```
  result<-c("EE"=list(result_EE$EE_uncoll),
                  "time_points"=list(time_points),
                  "maturity"=Ei,"sd"=list(result_EE$EE_uncollsd))
  if(Ei==1) result <-interpolate.grid.1Y(Ei,result_EE,
                              cashflow_frequency,time_points)
  return(result)
}

interpolate.grid.1Y<-function(Ei,result_EE, cashflow_frequency,
                                                time_points)
{
  exposure_values_raw<-result_EE$EE_uncoll
  time_points_raw<-time_points

  time_points_interp<-seq(cashflow_frequency,Ei,
                                    cashflow_frequency/2)
  exposure_values_interp<-approx(x=time_points_raw,
                                 y=exposure_values_raw,
                                 xout=time_points_interp)$y
  result<-c("EE"=list(exposure_values_interp),
                "time_points"=list(time_points_interp),
                                          "maturity"=Ei)
}

time_points.calc<-function(Ei,cashflow_frequency)
{
  nCashflows<-Ei%%cashflow_frequency
  if(nCashflows==0) iCashflow<-cashflow_frequency else
                              iCashflow<-nCashflows
  return(seq(iCashflow,Ei,by=cashflow_frequency))
}

CalcGridExposures<-function(grid)
{
  curve_list<-split(grid[,1:3],seq(nrow(grid)))
  names(curve_list)<-rep("curve_params",dim(grid)[1])
  curve_list<-lapply(curve_list,setNames,c("beta0","beta1","beta2"))
  nRow<-dim(grid)[1]
  result<-vector(mode="list",length = nRow)
  for(i in 1:nRow)
  {
    curve_params<-curve_list[i]$curve_params
    Ei<-grid[i,4]
    pay_leg_rate<-grid[i,5]
    result[[i]]<-IRS.exposure(curve_params,Ei,pay_leg_rate)
  }
  return(result)
}

maturity_standardise<-function(outputList)
{
  for( i in 1:length(outputList))
  {
    outputList[[i]]$time_points<-
        outputList[[i]]$time_points/outputList[[i]]$maturity
  }
  return(outputList)
}
```

A.4 OUTPUT GRID

```
beta.function.fit.area<-function(x,y,T)
{
  return(nlsLM(y~Const*beta(alpha,beta_param) *
                      x^(alpha - 1)*(T-x)^(beta_param-1),
                      start=list(Const=1, alpha=1, beta_param=1),
                      control = nls.lm.control(maxiter = 1024)))
}

beta.function.parameters<-function(EE)
{
  N<-length(EE)
  n<-3 # number of parameters in nls estimation of beta function
  result<-matrix(data=NA,nrow=N,ncol = n)
  for( i in 1:N)
  {
    ######fit beta function
    fit<-beta.function.fit.area(x=EE[[i]]$time_points,
                                y=EE[[i]]$EE,T=1)
    parameters<-c(summary(fit)$parameters[1,1],
                  summary(fit)$parameters[2,1],
                      summary(fit)$parameters[3,1])
    result[i,]= parameters
  }
  return(result)
}
```

A.5 BIPLOT INTERPOLATION FUNCTIONS

```
BiplotEE_estimate<-function(binN,test_input_bin)
{
  binMat<-binMaturities(binN)
  biplot_estimates_TxY<-biplotEstimates(binMat$Tx,test_input_bin)
  biplot_estimates_TyY<-biplotEstimates(binMat$Ty,test_input_bin)

  # adjust area under curve ...
  # const = area under curve * beta(a,b)

  biplot_estimates_TxY[,1]<-biplot_estimates_TxY[,1]*
                  beta(biplot_estimates_TxY[,2],
                  biplot_estimates_TxY[,3])
  biplot_estimates_TyY[,1]<-biplot_estimates_TyY[,1]*
                beta(biplot_estimates_TyY[,2],
                biplot_estimates_TyY[,3])

  xfits<-fits_all(test_input_bin,biplot_estimates_TxY)
  yfits<-fits_all(test_input_bin,biplot_estimates_TyY)
  result<-estimate_fits(test_input_bin,binMat$Tx,binMat$Ty,
                                        xfits,yfits)
  return(result)
}

biplotEstimates<-function(inputMaturity,test_input_bin)
{
  biplot_input<-readRDS(paste0("results\\biplot_data\\",
```

```
                                    inputMaturity,"Y_bipl.RDS"))
  input_plot_data<-readRDS(paste0("results\\biplot_data\\",
                           inputMaturity,"Y_inputData.RDS"))
  outputData<-readRDS(paste0("results\\biplot_data\\",
                          inputMaturity,"Y_outputData.RDS"))

  result<-biplot.interpolation.estimates_adj(biplot_input,
         input_plot_data[,-4],outputData,test_input_bin[,-4])
  return(result)
}

binMaturities<-function(binN)
{
  if(binN==1){return(list(Tx=1,Ty=5))}
  else if(binN==2) {return(list(Tx=5,Ty=10))}
  else if(binN==3){return(list(Tx=10,Ty=15))}
  else if(binN==4){return(list(Tx=15,Ty=20))}
  else if(binN==5) {return(list(Tx=20,Ty=25))}
  else if(binN==6){return(list(Tx=25,Ty=30))}
  else (return("invalid␣bin"))
}

biplot.interpolation.estimates_adj<-function(biplot_input,
                                inputData,outputData,testData)
{
  znew<-z.coordinates.star(inputData,testData,biplot_input)
  N<-dim(testData)[1]
  n<-dim(outputData)[2]
  result<-matrix(data=NA,nrow=N,ncol =n)
  for(i in 1:N)
  {
    outputGRIDestimates<-apply(outputData,2,biplot.interpolation,
                                inputData=inputData,
                                testData=testData,
                                biplot_input=biplot_input,
                                znew=znew[i,])
    result[i,]<-outputGRIDestimates
  }
  colnames(result)<-c("const","alpha","beta")
  return(result)
}

z.coordinates.star<-function(inputData,testData,bipl)
{
  xMeans <- apply(inputData,2,mean)
  xStdevs <- sqrt(apply(inputData,2,var))

  Zstar <- scale(testData,xMeans,xStdevs)%*%bipl$V[,1:2]
  Zstar <- as.data.frame(Zstar)
  colnames(Zstar) <- c("X1","X2")
  return(Zstar)
}

biplot.interpolation<-function(inputData,outputVec,testData,
                                            biplot_input,znew)
{
  ### extrap linear = FALSE, extrap = TRUE
  #result<-interp.new(linear=FALSE,extrap=TRUE,
                      biplot_input$Z[,1],biplot_input$Z[,2],
```

```
                              outputVec,znew[1],znew[2])$z
  result<-interp(linear=TRUE,extrap=FALSE,
                      biplot_input$Z[,1],biplot_input$Z[,2],
                               outputVec,znew[1],znew[2])$z
   return(result)
}

fits_all<-function(test_input, biplot_estimates)
{
  N<-dim(biplot_estimates)[1]
  result<-vector(mode="list",length = N)
  for(i in 1:N)
  {
    result[[i]]<-list_biplot_estimates_T1(list(T=test_input$T[i]),
                                        biplot_estimates[i,])
  }
  return(result)
}

estimate_fits<-function(test_input,Tx,Ty,biplot_fitsTx,
                                            biplot_fitsTy)
{
  N<-length(biplot_fitsTx)
  result<-biplot_fitsTx
  for(i in 1:N)
  {
    Tin<-test_input$T[[i]]
      wx<-1-(Tin-Tx)/(Ty-Tx)
      wy<-1-wx
    result[[i]]$fits<-(wx*biplot_fitsTx[[i]]$fits)+
                        (wy*biplot_fitsTy[[i]]$fits)
  }
  return(result)
}

list_biplot_estimates_T1<-function(test_input,biplot_estimates)
{
  ####set maturity =1
  maturity<-1
  x<-time_points.calc(test_input$T,0.25)/test_input$T
  y<-fit_beta(maturity,x,biplot_estimates[1],
                  biplot_estimates[2],biplot_estimates[3])
  list_result<-list(time_points=x,fits=y)
  return(list_result)
}
```

A.6 ERROR CALCULATIONS

```
error_mat<-function(sdN,bin_data,binN, true_testInput,biplot_estY)
{
  n<-length(biplot_estY)
  result<-matrix(data=NA,nrow=n,ncol = 1)
  for(i in 1:n)
  {

    ind_bin_sample<-which(bin_data[1,binN] <=test_input[,4]  &
                        test_input[,4]<= bin_data[2,binN])
```

```
    iSampleBin<-ind_bin_sample[i]
    true_Tin<-true_testInput[[iSampleBin]]
    lower<-true_Tin$EE-sdN*true_Tin$sd
    upper<-true_Tin$EE+sdN*true_Tin$sd
    result[i,]<-1-sum(cbind(lower <= biplot_estY[[i]]$fits &
    biplot_estY[[i]]$fits<= upper )) /length(biplot_estY[[i]]$fits)
  }
  return(result)
}

errors_all<-function(bin_data,binN,true_testInput,
                                                 biplot_estY)
{
  result<-matrix(data = NA,nrow = 1,ncol = 3)
    for(i in 1:dim(result)[2])
    {
      sdN<-c(0.5,1,1.5)
      err<-error_mat(sdN[i],bin_data,binN, true_testInput,
                                                 biplot_estY)
      result[,i]<-round(mean(err[!is.na(err)]),4)
  }
  return(result)
}
```

A.7 TEST INPUT

```
inputGRID<-readRDS("results\\inputSwap.RDS")
inputNS<-readRDS("results\\HistNSParams.rds")
outn <- 100
gridminmax <- rbind(apply(inputGRID,2,min), apply(inputGRID,2,max))
gridminmax[,4] <- c(1,30)

#  sample from historical NS parameters
set.seed(1258)

NSparamsSampleInput<-as.data.frame(inputNS[sample(dim(inputNS)[1],
                                                        outn),1:3])
historicalNSData <- as.data.frame(cbind(runif(outn,gridminmax[1,4],
                                   gridminmax[2,4]),
                                   runif(outn,gridminmax[1,5],
                                   gridminmax[2,5])))
historicalNSData <-cbind(NSparamsSampleInput,outofsampleINPUT)
historicalNSData <- outofsampleINPUT[order(outofsampleINPUT[,4]),]


# within-grid data
set.seed(1258)
WithinGridData <- as.data.frame(cbind(runif(outn,gridminmax[1,1],
                                                 gridminmax[2,1]),
                                    runif(outn,gridminmax[1,2],
                                                 gridminmax[2,2]),
                                    runif(outn,gridminmax[1,3],
                                                 gridminmax[2,3]),
                                    runif(outn,gridminmax[1,4],
                                                 gridminmax[2,4]),
                                    runif(outn,gridminmax[1,5],
                                                 gridminmax[2,5])))
```

```
WithinGridData <- inSampleINPUT[order(inSampleINPUT[,4]),]
```

A.8 PLOT FUNCTIONS

A.8.1 Contour plots

```
biplot_input<-readRDS(paste0("results\\biplot_data\\",
                                    inputMaturity,"Y_bipl.RDS"))
input_plot_data<-readRDS(paste0("results\\biplot_data\\",
                                 inputMaturity,"Y_inputData.RDS"))
outputData<-readRDS(paste0("results\\biplot_data\\",
                                 inputMaturity,"Y_outputData.RDS"))

interp.out <- interp(biplot_input$Z[,1],biplot_input$Z[,2],
                                                 outputData[,2])
contour(interp.out,col=brewer.pal(n=9,"PuBuGn"),nlevels = 9,
                                                     add = TRUE)
```

A.8.2 Biplot interpolation results output

```
 results_plot<-function(binN,true_Tin,iSample,biplot_estY)
{
  bin_data<-matrix(data=c(1,5,5,10,10,15,15,20,20,25,25,30),
                                         nrow=2,byrow = FALSE)
  ind_bin_sample<-which(bin_data[1,binN] <=test_input[,4]  &
                             test_input[,4]<= bin_data[2,binN] )
  iSampleBin<-ind_bin_sample[iSample]

  true_Tin<-true_testInput[[iSampleBin]]

  m<-matrix(c(1,2,3,4),nrow=4, ncol=1,byrow=TRUE)
  layout(mat=m,heights=c(0.3,0.3,0.3,0.1))


  par(mar = c(2.9,2.9,1,1))
  par(mgp=c(2,1,0))

  for(sdN in c(0.5,1,1.5))
  {
    maxY<-max(true_Tin$EE+sdN*true_Tin$sd,biplot_estY[[iSample]]$fits)

    plot(ylim=c(0,maxY),true_Tin$time_points,true_Tin$EE,
                        col="black",xlab="Maturity",
                                ylab=paste0("EE␣(␣",sdN,"␣SD␣)"),
                                                        cex.lab=0.7)
    lines(true_Tin$time_points,true_Tin$EE,col="black")

    for(i in c(1,-1))
    {
      points(true_Tin$time_points,true_Tin$EE+i*sdN*true_Tin$sd,
                                                      col="grey")
      lines(true_Tin$time_points,true_Tin$EE+i*sdN*true_Tin$sd,
                                            col="grey", lty=2)
    }

    Tin<-true_Tin$maturity
```

```
    points(biplot_estY[[iSample]]$time_points*Tin,
                    biplot_estY[[iSample]]$fits,col="red")
    lines(biplot_estY[[iSample]]$time_points*Tin,
                    biplot_estY[[iSample]]$fits,col="red")
  }

  par(mar=c(0,0,0,0))
  plot(1, type = "n", axes=FALSE, xlab="", ylab="")
  legend(box.lty = 0,horiz = TRUE,x="center",inset=0,
                    legend=c("EEest","EEtrue","SD"),
                            col=c("red","black","grey"),
                                    lty=c(1,1,2),cex=0.7)
}
```